Apprentissage des préférences humaines pour la préhension robotique

Yoann Fleytoux

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Apprentissage des préférences humaines pour la préhension robotique

THÈSE

présentée et soutenue publiquement le 01/12/2023

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Fleytoux Yoann

Composition du jury

Rapporteurs :	M Sylvain Calinon	Senior Research Scientist, Idiap, Switzerland
	M Youcef Mezouar	Professeur, Institut Pascal-Université Clermont Auvergne
Examinateurs :	Mme Bernadetta Addis	Maîtresse de conférences, Université de Lorraine, Loria, Nancy
	M Christophe Leroux	Manager European Affairs in AI and robotics, CEA, France
Encadrants :	M Jean-Baptiste Mouret	Directeur de Recherche, Inria Nancy Grand-Est
	Mme Serena Ivaldi	Directrice de Recherche, Inria Nancy Grand-Est

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Remerciements

Dans cette section de remerciements, je tiens tout d'abord à exprimer ma profonde gratitude envers Serena et Jean-Baptiste, qui m'ont encadré tout au long de ce travail et m'ont apporté un soutien inestimable.

Je tiens également à remercier l'Istituto Italiano di Tecnologia qui m'a accueilli lors de mon échange, ainsi que Lorenzo, Alessandro et en particulier Fabrizio pour leur assistance technique et humaine. Vous avez tous contribué à enrichir mon expérience et à renforcer mes compétences.

Un merci tout particulier à Adrien Guénard pour son travail au sein du Créativ' Lab et pour ses précieux conseils techniques qui ont été d'une grande aide.

Je voudrais également remercier l'ensemble de l'équipe LARSEN, au sein de laquelle j'ai eu l'opportunité d'effectuer ma thèse. Votre dynamisme et votre amitié ont été essentiels pour mener à bien mes recherches.

Je tiens à remercier Waldez, Luigi et Jessica pour le temps que nous avons passé ensemble lors des voyages. Ce fut une expérience mémorable et enrichissante.

Je tiens à souligner l'importance du travail de tous les membres du personnel du laboratoire, notamment les techniciens, le personnel de nettoyage, la cantine, la cafétéria et les administratifs. Votre dévouement quotidien a créé un environnement propice à la réussite de ma thèse. Je tiens également à remercier les personnes rencontrées dans le cadre plus large du laboratoire, notamment Oriane et Florian.

Je suis également reconnaissant envers Thaïs, Léonie et Corentin avec lesquels j'ai eu le plaisir de travailler lors de leur stage.

Un grand merci à Mihai, Anji et Lorenzo, avec qui j'ai collaboré au Créativ'Lab. Votre soutien et votre amitié ont été précieux tout au long de cette aventure.

Je remercie ma famille et mes amis proches, qui m'ont soutenu durant ces années. Merci pour les JDL, les repas, le sport, les balades en centre ville accompagnées de Friedrich et Karl.

Enfin, je remercie Solène pour sa patience et son soutien inconditionnel tout au long de cette période. Ton soutien a été essentiel pour m'aider à surmonter les moments difficiles.

Je tiens à préciser que je ne remercie pas le COVID.

Sommaire

Table des figures

Résumé

Cette thèse aborde le défi de la préhension d'objets aux prises difficiles par des robots manipulateurs en combinant l'apprentissage automatique, l'utilisation de capteurs tactiles et l'expertise humaine. Dans certaines applications industrielles telles que la manipulation de déchets, la téléopération de robots est utilisée pour manipuler des objets irréguliers. Cette tâche est complexe à automatiser en raison de l'irrégularité, de la visibilité partielle, de la fragilité et de la susceptibilité des objets à se briser lors de la manipulation ou du transport. Dans ce contexte, l'expertise de l'opérateur est cruciale pour guider les robots, car ses critères de décision ne se traduisent pas facilement en critères mathématiques implémentables dans un algorithme automatique. Apprendre à partir d'exemples est une approche pertinente. Cependant, étant donné que ces démonstrations sont coûteuses à acquérir, les travaux de cette thèse proposent des pistes nécessitant peu de démonstrations de l'expert.

L'objectif est d'améliorer les compétences de préhension du robot en utilisant l'apprentissage hors ligne et en ligne, ce qui permet au robot d'apprendre rapidement les préférences de préhension adaptées aux objets en question. Ainsi, cette thèse contribue au développement de solutions robotiques avancées pour la manipulation d'objets complexes.

Mots-clés: Apprentissage automatique, Robotique, Manipulation

Abstract

This thesis addresses the challenge of gripping objects difficult to handle by manipulator robots by combining machine learning, the use of tactile sensors, and human expertise. In certain industrial applications such as waste handling, robot teleoperation is used to manipulate irregular objects. This task is complex to automate due to the irregularity, partial visibility, fragility, and susceptibility of the objects to break during manipulation or transportation. In this context, the operator's expertise is crucial in guiding the robots, as their decision criteria do not easily translate into mathematical criteria implementable in an automatic algorithm. Learning from examples is a relevant approach. However, since these demonstrations are costly to acquire, the work in this thesis proposes directions that require few expert demonstrations.

The goal is to improve the robot's gripping skills using offline and online learning, which allows the robot to quickly learn the gripping preferences suitable for the objects in question. Thus, this thesis contributes to the development of advanced robotic solutions for the manipulation of complex objects.

Keywords: Machine Learning, Robotics, Grasping.

Chapitre 1

Introduction

La robotique est un domaine en constante évolution [1] . Elle transforme la manière dont les industries fonctionnent et dont les humains interagissent avec les machines. Avec l'avènement de la quatrième révolution industrielle, l'intelligence artificielle et la robotique sont au cœur des changements qui visent à développer des systèmes autonomes et intelligents [164]. La manipulation d'objets par des robots est un enjeu majeur de la recherche en ce domaine, c'est une tâche essentielle pour de nombreuses applications.

La communauté des chercheurs en robotique s'efforce de développer des robots capables de fonctionner dans des environnements non structurés et dynamiques, tels que les chantiers de construction, les scènes de catastrophe, les missions spatiales ou même une simple maison ou un supermarché [2]. Ces environnements posent des défis uniques aux robots car ils impliquent souvent un terrain imprévisible, des conditions changeantes et des obstacles inattendus. Les robots capables de saisir des objets de manière autonome peuvent être utilisés pour automatiser des tâches répétitives et dangereuses, ce qui peut améliorer la sécurité et la productivité dans de nombreuses industries. Ils peuvent également être utilisés pour aider les personnes âgées et les personnes handicapées dans leur vie quotidienne [131].

En apprenant à saisir des objets, les robots peuvent les prendre, les déplacer et interagir avec leur environnement de manière plus sophistiquée, leur permettant d'effectuer des tâches plus complexes ou de remplacer l'humain dans des cas où l'utilisation de robots est essentielle.

Dans la littérature sur la robotique et la manipulation, le domaine de la préhension est souvent considéré comme mature. En effet, les études menées depuis plusieurs décennies ont proposé une vaste gamme de représentations, méthodes et mesures de la qualité de la préhension, calculées à partir d'informations sur les mouvements de la main ou de la pince et sur la forme de l'objet lorsqu'elle est connue [86].

1. Vidéo 'Robots - A 50 Year Journey' de Oussama Khatib (2000) disponible à l'adresse `https://vimeo.com/137042620`

2. Vidéo 'Robots - The Journey Continues' de Bruno Siciliano, Oussama Khatib et Torsten Kröger (2016) disponible à l'adresse `https://vimeo.com/173394878`

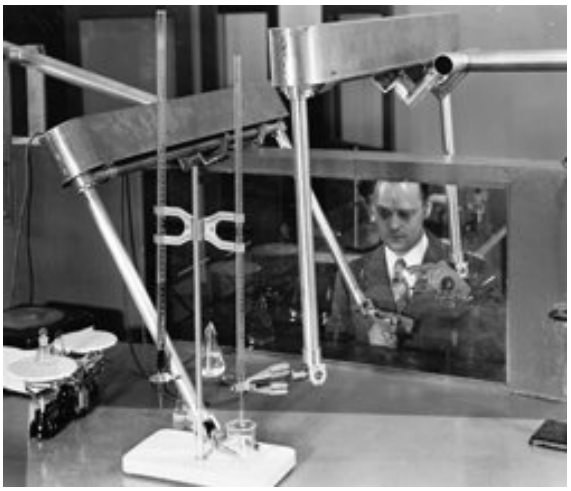


FIGURE 1.1 – Goertz faisant la démonstration de son manipulateur mécanique esclave-maître en 1949.

Historiquement, l'un des premiers exemples de manipulation télé-opérée date de 1949 (Fig. 1.1). Alors qu'il travaillait pour la Commission de l'énergie atomique au Laboratoire national d'Argonne, Raymond C. Goertz a déposé un brevet [59] pour un premier manipulateur maître-esclave afin de manipuler des matériaux radioactifs. Il a mis au point un système composé de deux parties : une console maître, actionnée par un humain, et un dispositif esclave, qui était la machine à contrôler. La console maître était reliée au dispositif esclave par une liaison de communication, ce qui permettait à l'opérateur d'envoyer des commandes à la machine et de recevoir un retour d'information sur ses actions. Le dispositif esclave était équipé de capteurs et d'actionneurs qui lui permettaient d'exécuter les commandes de l'opérateur et de renvoyer des informations à la console principale. Afin d'expérimenter en toute sécurité avec des matériaux radioactifs, il est nécessaire de minimiser autant que possible l'effet de la radioactivité nocive, comme les rayons gamma.

Cette nécessité a favorisé la mise au point de nombreux appareils télécommandés permettant à l'expérimentateur d'effectuer ses opérations indirectement, derrière un mur de protection, en observant le résultat de ses mouvements à travers un jeu de miroirs ou une fenêtre blindée dans le mur. Les difficultés relevées par cet exemple sont toujours d'actualité [172] et font partie des défis adressés dans cette étude.

Le tri semi-autonome de tas d'objets inconnus est une tâche ayant de multiples applications potentielles répondant à d'énormes besoins sociétaux, comme le tri de déchets radioactifs ou d'autres matières dangereuses. Par exemple, le nettoyage du demi-siècle de déchets radioactifs accumulés au Royaume-Uni (principalement sur le site de Sellafield) représente l'un des plus grands projets de réhabilitation environnementale en Europe, avec des coûts prévus de plus de 50 milliards d'euros au Royaume-Uni seul au cours des 100 prochaines années [115].

De nombreux anciens sites nucléaires de l'UE (plus de 60 ans au Royaume-Uni) contiennent un grand nombre de conteneurs de stockage, dont beaucoup ont des contenus inconnus de niveaux de contamination mixtes. D'autres applications, comme la robotique dans la fabrication ou l'agriculture, rencontrent des défis similaires, où un tas d'objets inconnus ou déformés, comme différents types de récoltes, doit être manipulé [131].

FIGURE 1.2 – L'Unimate était le premier robot industriel jamais construit. C'était un bras manipulateur hydraulique qui pouvait effectuer des tâches répétitives.

Peu après, le premier robot véritablement programmable a été l'Unimate (Fig. 1.2), mis au point par George Devol et Joseph Engelberger en 1954. L'Unimate [3] a été utilisé pour automatiser la chaîne de production de General Motors et son exemple a ouvert la voie au développement des robots industriels modernes. UNIMATE obéissait à des commandes étape par étape stockées sur un tambour magnétique et, avec son bras de plus de 1800 kg, il a été utilisé par les constructeurs automobiles pour automatiser les processus de travail des métaux et de soudage. Ce robot a ouvert la voie au développement des robots industriels modernes.

Les robots industriels traditionnels sont généralement conçus pour effectuer des tâches spécifiques et sont souvent coûteux et difficiles à programmer. Les cobots, ou robots collaboratifs, ont été introduits dans les années 1990 [44] pour offrir de nouvelles possibilités d'interaction et de coopération entre humains et machines. Les cobots sont des robots légers et faciles à programmer qui peuvent fonctionner aux côtés des humains sans danger, ce qui en fait une option attractive pour de nombreuses applications.

La robotique collaborative, ou cobotique, offre également de nouvelles possibilités pour l'interaction et la coopération entre les humains et les robots dans la saisie d'objets [10]. Les cobots sont conçus pour travailler aux côtés des humains, ce qui permet aux humains et aux robots de travailler ensemble pour accomplir des tâches de manière plus efficace et plus sûre. Les cobots peuvent être utilisés pour aider les humains à saisir des objets lourds ou encombrants, ou pour effectuer des tâches de saisie difficiles ou dangereuses.

Parallèlement à ces développements mécatroniques, l'état de l'art en matière de planification de saisie et d'apprentissage automatique a connu des avancées significatives au

3. Vidéo disponible à l'adresse `https://youtu.be/VdolSBpyCaU`

cours des dernières décennies [206]. Ces avancées sont dues à la plus grande disponibilité des données, à l'augmentation de la puissance de calcul des processeurs et à la disponibilité de capteurs pour améliorer la performance des robots. Cela permet aux robots de s'entraîner à partir d'exemples et de données sensorielles pour améliorer leur performance. L'apprentissage automatique peut être utilisé pour apprendre à saisir des objets inconnus en analysant les propriétés des objets et en reconnaissant les caractéristiques communes des objets dans des bases de données d'apprentissage. Les techniques de vision par ordinateur peuvent également être utilisées pour aider les robots à identifier les objets et à déterminer la meilleure façon de les saisir.

Les capteurs visuels, tels que les caméras RGB-D, sont un type de capteur clé utilisé en robotique pour des tâches telles que la reconnaissance, la localisation et la saisie d'objets [159]. Les caméras RGB-D fournissent à la fois des informations de couleur et de profondeur, permettant aux robots de percevoir et de comprendre leur environnement de manière plus détaillée et nuancée. Les capteurs tactiles peuvent aussi être utilisés pour aider les robots à saisir des objets en temps réel. Ils peuvent par ailleurs détecter les forces et les pressions appliquées lors de la saisie des objets, ce qui permet aux robots de s'adapter aux changements dans l'environnement et de saisir efficacement les objets. Les chercheurs ont également exploré l'utilisation de capteurs de force pour améliorer la planification de saisie, en mesurant la force appliquée lors de la saisie des objets et en utilisant cette information pour ajuster les mouvements du robot [191].

La planification de la préhension en robotique est le processus qui consiste à déterminer comment un robot doit saisir et manipuler un objet. Il s'agit de sélectionner la pose de préhension la plus appropriée, c'est-à-dire la position et l'orientation du préhenseur par rapport à l'objet, en fonction de la forme, de la taille et d'autres propriétés physiques de l'objet. L'objectif est de trouver une préhension qui soit à la fois stable et efficace, ce qui signifie qu'elle peut saisir et manipuler l'objet de manière fiable sans le faire tomber ou l'endommager.

Un "pipeline de préhension" est une séquence d'étapes qu'un robot suit pour saisir un objet [86]. Il comprend généralement quatre étapes : l'acquisition de la cible, la planification de la saisie, la planification du mouvement et l'exécution de la saisie. Le robot doit d'abord détecter l'objet et analyser sa forme, sa taille et sa position dans l'environnement. Une fois que l'objet a été perçu, le robot doit déterminer la meilleure façon de le saisir en fonction de sa forme, de sa taille et d'autres propriétés. Ensuite, le robot doit planifier une trajectoire pour son bras et sa pince afin de saisir l'objet et exécuter la saisie. Enfin, le robot doit exécuter la saisie planifiée en fermant la pince autour de l'objet.

Cependant, la saisie d'objets réguliers et irréguliers reste un défi, en raison de la diversité des formes et des matériaux des objets. L'intégration de la connaissance humaine peut également aider à améliorer la planification de saisie des robots [169]. La capacité des robots à manipuler de manière totalement autonome des tas denses comprenant des objets inconnus est limitée en raison des défis posés par la compréhension de la scène et donc de la prédiction de saisie. Les humains ont une grande expertise dans la préhension des objets, et cette connaissance peut être utilisée pour aider les robots à les saisir efficacement. Les chercheurs ont exploré l'utilisation de l'interaction humain-robot pour améliorer la planification de saisie, en permettant aux humains de fournir des informations sur les objets à saisir [26].

La thèse présentée ici explore cette approche pour améliorer la planification de saisie

en robotique. Elle s'appuie d'une part, sur la connaissance humaine et d'autre part, sur l'emploi de capteurs tactiles dans le contexte de la manipulation d'objets inconnus. Dans les travaux de cette thèse, nous considérons une approche semi-autonome où un opérateur humain peut interagir avec le système (par exemple en utilisant la télé-opération) et donner des commandes de haut niveau pour à terme atteindre l'exécution autonome des saisies. Le système assimile des compétences de manipulation à partir des exemples fournis par l'opérateur humain et peut ainsi apprendre à saisir de manière efficace en terme de données. Nous abordons le problème de la saisie d'objets inconnus, c'est-à-dire que nous ne disposons pas d'un modèle de l'objet et la caméra observant la scène n'offre qu'une vision en plongée des objets.

Plus précisément, cette thèse cherche à répondre aux questions suivantes :

— Comment intégrer la connaissance humaine dans des algorithmes de planification de saisie existant pour améliorer leur performance et leur adaptabilité ?
— Comment utiliser des capteurs tactiles pour améliorer la précision et la stabilité de la saisie d'objets inconnus ?
— Quelles sont les stratégies et les défis rencontrés par les humains lors de la manipulation d'objets varié ?
— Comment ces informations peuvent-elles être utilisées pour améliorer l'apprentissage automatique en robotique ?

Pour répondre à ces questions, nous avons mené une série d'expériences et d'études, des approches d'apprentissage automatique et l'analyse de données collectées auprès d'opérateurs humains. Nous avons également développé et comparé différents algorithmes de planification de saisie pour évaluer leur efficacité et leur polyvalence, en simulation et avec des expériences réelles.

L' annexe A décrit l'environnement matériel et logiciel utilisé dans cette thèse, ainsi que la collecte et le traitement des données. Nous décrivons les capteurs tactiles utilisés pour la saisie d'objets. Nous décrivons également les algorithmes de planification de saisie que nous avons développés et comparés.

Le chapitre 2 présente l'état de l'art en matière de planification de saisie et d'apprentissage automatique en robotique, ainsi que les techniques d'interaction humain-robot. Nous passons en revue les approches actuelles pour la planification de saisie, en mettant l'accent sur les avantages et les limites de chacune. Nous discutons également des techniques d'apprentissage automatique pour la saisie d'objets, y compris l'apprentissage par renforcement, l'apprentissage supervisé et l'apprentissage non supervisé.

Le chapitre 3 détaille l'intégration de la connaissance humaine dans le pipeline de préhension de manière à respecter certaines propriétés de l'objet. L'essentiel du travail réalisé se concentre donc sur les parties acquisition de la cible et la planification de la saisie. Nous nous intéressons à des cas où l'expertise humaine permet de respecter des contraintes que l'on ne peut pas déduire seulement des données fournies par la caméra. Par exemple, nous ne voulons pas endommager les parties fragiles de l'objet, comme les verres des lunettes. De même, nous ne voulons pas saisir les parties coupantes d'un objet pour éviter d'endommager le préhenseur. Ou encore, nous devons parfois tenir compte d'une répartition inégale de la masse pour saisir l'objet de façon à ce qu'il ne s'incline pas ou ne tombe pas de la pince, comme dans le cas du marteau (Fig. 1.3).

FIGURE 1.3 – Exemples de préférences.

Dans le chapitre 4, nous étudierons une application des capteurs tactiles, notamment l'intégration de capteurs tactiles sur les pinces d'un robot afin de mesurer la pression et la déformation de l'objet saisi. Nous aborderons le problème de la saisie d'objets inconnus identifiés à partir d'images en vue de dessus avec un préhenseur parallèle. Les générateurs de préhension actuels déterminent les meilleurs emplacements candidats pour les préhensions planaires en utilisant une image RGB-D, mais ne parviennent pas à obtenir des résultats satisfaisants lorsque les objets sont fins, transparents ou présentent des formes courbes. Nous proposons d'entraîner un régresseur capable de prédire la meilleure hauteur de saisie à partir de l'image. Pour ce faire, nous utilisons un jeu de données acquis automatiquement grâce aux capteurs optiques tactiles du DIGIT, permettant d'évaluer le succès et la stabilité de la prise.

Dans le chapitre 5, nous examinerons les performances de saisie d'une vingtaine de participants manipulant un tas d'objets variés. L'objectif de cette étude est de fournir une base de référence des performances humaines dans la résolution de ce problème, ainsi que de tirer des conclusions sur les stratégies adoptées et l'expérience de manipulation en elle-même. Au cours de cette analyse, nous observerons attentivement les approches et techniques utilisées par les participants pour saisir efficacement les objets. Cette évaluation nous permettra d'identifier les facteurs qui contribuent à une meilleure performance, ainsi que les défis rencontrés lors de la manipulation d'objets dans un environnement complexe. Les résultats obtenus serviront de pistes d'améliorations des algorithmes de préhension robotique afin de concevoir des évaluations pertinentes et adaptées aux défis posés par la manipulation d'objets divers et inconnus.

Enfin, dans le chapitre 6, nous réaliserons une comparaison approfondie de différents algorithmes de référence utilisés pour la planification de saisie, en étudiant leur efficacité et leur adaptabilité. Cette comparaison nous permettra d'évaluer les forces et les faiblesses de chaque algorithme, ainsi que d'identifier les meilleures pratiques pour optimiser la planification de saisie robotique via la mise en œuvre dans une version interactive des algorithmes décrits précédemment au chapitre 3. La version interactive (en ligne) devrait améliorer la performance et en s'adaptant aux préférences et aux contraintes.

En résumé, cette thèse explore de nouvelles approches pour améliorer la planification de saisie en robotique en tirant parti de la connaissance humaine et de capteurs tactiles. Les résultats obtenus contribuent à une meilleure compréhension des défis et des stratégies associés à la manipulation d'objets, ainsi qu'à l'amélioration des performances des robots pour une grande variété d'applications, notamment dans le domaine du tri des déchets

nucléaires. L'apprentissage automatique et l'analyse des performances de saisie humaine peuvent aider à résoudre les problèmes de saisie d'objets inconnus en robotique, tandis que l'utilisation de capteurs tactiles peut améliorer la précision et la stabilité de la saisie.

1.1 Contexte - European project HEAP

Cette thèse s'inscrit dans le projet HEAP financé par Chist-Era, qui vise à développer des algorithmes de manipulation robotique pour le tri des déchets nucléaires. Ce projet est mené en partenariat avec l'Université de Lincoln, l'Institut Italien de Technologie (IIT), l'IDIAP et l'Université de Vienne, et aborde plusieurs scénarios de tâches de complexité variable, tels que saisir et pousser des objets irréguliers, sélectionner des objets dans un tas, identifier toutes les instances d'objets et les trier.

Dans le cadre de ce projet, j'ai effectué un séjour de près de 3 mois à l'Institut Italien de Technologie, où j'ai collaboré avec l'équipe de Lorenzo Natale pour intégrer mes algorithmes dans leur outil d'évaluation des performances des algorithmes de saisie.

1.2 Contributions

L'ensemble des travaux de cette thèse a produit plusieurs contributions sous forme d'articles scientifiques décrits ci-dessous.

1.2.1 Articles acceptés / publiés

Résumé : La saisie a fait des progrès impressionnants au cours des dernières années grâce à l'apprentissage profond. Cependant, il existe de nombreux objets pour lesquels il n'est pas possible de choisir une préhension en regardant uniquement une image RGB-D, que ce soit pour des raisons physiques (par exemple, un marteau avec une distribution de masse inégale) ou des contraintes de tâche (par exemple, de la nourriture qui ne doit pas être gâchée). Dans de telles situations, les préférences des experts doivent être prises en compte.

Dans cet article, nous présentons un pipeline de préhension efficace en termes de données (Latent Space GP Selector — LGPS) qui apprend les préférences de préhension avec seulement quelques étiquettes par objet (typiquement 1 à 4) et qui se généralise à de nouvelles vues de cet objet. Notre pipeline est basé sur l'apprentissage d'un espace latent de préhension à partir d'une base de données généré par un générateur de préhension de pointe (par exemple, Dex-Net). Cet espace latent est ensuite utilisé comme une entrée de faible dimension pour un classificateur à processus gaussien qui sélectionne la saisie préférée *parmi celles proposées par le générateur.*

Les résultats montrent que notre méthode surpasse à la fois GR-ConvNet et GG-CNN (deux méthodes de pointe qui sont également basées sur des prises étiquetées) sur le jeu de données Cornell, en particulier lorsque seulement quelques étiquettes sont utilisées : seulement 80 étiquettes sont suffisantes pour choisir correctement 80% des prises (885 scènes, 244 objets). Les résultats sont similaires sur notre jeu de données (91 scènes, 28 objets).

(IEEE ICRA 2022) - Data-efficient learning of object-centric grasp preferences
Yoann Fleytoux, Anji Ma, Serena Ivaldi, Jean-Baptiste Mouret

Résumé : Nous abordons le problème de la saisie d'objets inconnus identifiés à partir d'images descendantes avec un préhenseur parallèle. Lorsqu'aucun modèle 3D de l'objet n'est disponible, les générateurs de préhension de l'état de l'art identifient les meilleurs emplacements candidats pour les préhensions planaires en utilisant l'image RGB-D. Cependant, alors qu'ils génèrent la position cartésienne et l'orientation de la pince, la hauteur du centre de la prise est souvent déterminée par une heuristique basée sur le point le plus haut de la carte de profondeur, ce qui conduit à des prises infructueuses lorsque les objets ne sont pas épais, ou ont des transparences ou des formes courbes. Dans cet article, nous proposons d'apprendre un régresseur qui prédit la meilleure hauteur de saisie à partir de l'image. Nous entraînons ce régresseur à l'aide d'un jeu de données acquis automatiquement grâce aux capteurs optiques tactiles du DIGIT, qui peuvent évaluer la réussite et la stabilité de la saisie. En utilisant notre prédicteur, le succès de la saisie est amélioré de 6% pour tous les objets, de 16% en moyenne sur les objets difficiles, et de 40% pour les objets qui sont notamment très difficiles à saisir (par exemple, transparents, courbés, minces). [4]

(IEEE ICRA 2023) - Learning height for top-down grasps with the DIGIT sensor
Thais Bernardi*, **Yoann Fleytoux***, Jean-Baptiste Mouret, Serena Ivaldi

Résumé : Les modèles de prédiction visuelle constituent une solution prometteuse pour la saisie robotique visuelle d'objets mous inconnus et encombrés. Les modèles précédents de la littérature sont gourmands en calcul, ce qui limite la reproductibilité ; bien que certains considèrent la stochasticité dans le modèle de prédiction, elle est souvent trop faible pour saisir la réalité des expériences robotiques impliquant la saisie de tels objets. De plus, les travaux précédents se sont concentrés sur des mouvements élémentaires qui ne sont pas efficaces pour raisonner en termes d'actions sémantiques plus complexes. Pour remédier à ces limitations, nous proposons VP-GO, un modèle de prédiction visuelle stochastique "léger" conditionné par l'action. Nous proposons une décomposition hiérarchique des actions sémantiques de préhension et de manipulation en mouvements élémentaires de l'effecteur final, afin d'assurer la compatibilité avec les modèles et les ensembles de données existants pour la prédiction visuelle d'actions robotiques tels que RoboNet. Nous enregistrons et diffusons également un nouveau jeu de données ouvert pour la prédiction visuelle de la saisie d'objets, appelé PandaGrasp. Notre modèle peut être pré-entraîné sur RoboNet et affiné sur PandaGrasp, et ses performances sont similaires à celles de modèles plus complexes en termes de métriques de prédiction du signal. Qualitativement, il est plus performant lorsqu'il s'agit de prédire le résultat de saisies complexes effectuées par notre robot.

4. * Les deux premiers auteurs de l'article scientifique en question ont contribué à parts égales à la recherche et à la rédaction de l'article.

(IEEE ARM 2022) - VP-GO : A 'Light' Action-Conditioned Visual Prediction Model for Grasping Objects
Anji Ma, **Yoann Fleytoux**, Jean-Baptiste Mouret, Serena Ivaldi

1.2.2 Articles en cours

Résumé :
La saisie robotique robuste d'objets a de vastes applications industrielles. La fiabilité des méthodes de préhension basées sur des données est influencée par la variabilité des formes d'objets rencontrées pendant la formation. La plupart des ensembles de données d'objets existants souffrent d'un biais de sélection humaine, manquent de variabilité ou ne sont pas reproductibles. Cet article présente un ensemble de données d'objets imprimables en 3D physiquement reproductibles pour l'entraînement et l'évaluation des algorithmes de préhension. Il contient les maillages 3D exacts de 50 objets à des fins de simulation et d'impression. Les différents objets du jeu de données ont été trouvés à l'aide de l'algorithme MAP-Elites, qui optimise la variabilité des objets en fonction de deux métriques de préhension. Nous avons utilisé un Variational AutoEncoder (VAE) comme modèle génératif pour les modèles d'objets voxelgrid, qui ont ensuite été convertis en mailles et simplifiés à l'aide de la Volumetric Hierarchical Approximate Convex Decomposition (V-HACD). L'ensemble de données est accessible au public en ligne et peut être commandé auprès de n'importe quel service d'impression 3d selon des spécifications données. Nous espérons qu'il deviendra un ensemble de données d'évaluation standard pour la communauté de la préhension robotique.

AGOD-Grasp : an Automatically Generated Object Dataset for benchmarking and training robotic grasping algorithms
Mihai Andries, **Yoann Fleytoux**, Jean-Baptiste Mouret, Serena Ivaldi

Chapitre 2

État de l'art

Le problème de la synthèse de préhension robotique est complexe, car il englobe de multiples variables et contraintes liées à la tâche que le robot doit accomplir. La résolution de ce problème est cruciale pour permettre aux robots d'interagir efficacement avec leur environnement, notamment dans des tâches telles que la manipulation d'objets pour l'assemblage/désassemblage ou le transport d'objets.

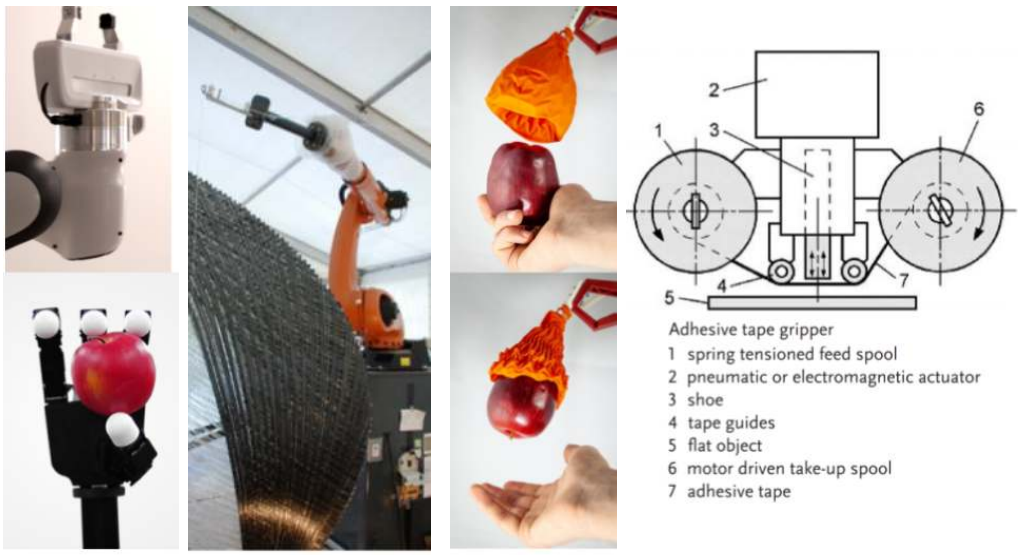
Dans un premier temps, nous aborderons les caractéristiques essentielles du problème de la synthèse de préhension robotique (Sec. 2.1), ainsi que les représentations des préhensions couramment utilisées en robotique (Sec. 2.2). Par la suite, nous présenterons plusieurs approches pour résoudre ce problème, en nous concentrant notamment sur les méthodes géométriques (Sec. 2.4) et les méthodes basées sur l'apprentissage automatique (Sec. 2.5). Chacune de ces approches présente des avantages et des limites spécifiques, et le choix de la méthode appropriée dépendra des exigences particulières de l'application en question. Ainsi, il est essentiel de bien comprendre les différentes techniques disponibles pour choisir la solution la mieux adaptée au contexte et aux objectifs visés.

2.1 Caractéristiques du problème de préhension

La recherche d'une prise adéquate parmi l'ensemble infini de candidats possibles est un problème complexe qui a été fréquemment abordé dans la communauté robotique, engendrant ainsi une multitude d'approches variées. Plusieurs critères peuvent être utilisés pour distinguer ces méthodes [11, 86, 11, 206], parmi lesquels nous avons sélectionné les éléments suivants : le type d'effecteur utilisé (Sec. 2.1.1), les caractéristiques des objets à saisir (Sec. 2.1.2), la distinction entre méthodes visant à saisir des objets isolés ou en groupe (Sec. 2.1.3), la synthèse de préhensions avec 4 ou 6 degrés de libertés (Sec. 2.1.4), l'utilisation ou non d'un modèle des objets (Sec. 2.1.5), la capacité à offrir des performances en temps réel (Sec. 2.1.6) et, enfin, la prise en compte d'actions non préhensibles avant la manipulation (Sec. 2.1.7).

En examinant ces critères, nous pouvons mieux comprendre et comparer les différentes approches proposées pour résoudre le problème de la synthèse de préhension robotique. Cette analyse permettra d'identifier les méthodes les mieux adaptées aux exigences spécifiques de chaque situation et d'orienter les recherches futures pour améliorer les performances et l'efficacité des robots dans diverses tâches de manipulation.

2.1.1 Types d'effecteurs



(a) (b) (c) (d)

FIGURE 2.1 – Exemples de différents types de préhenseurs : (a) Préhenseurs à contact mécanique (impactive gripper) : dispositifs tels que mâchoires ou griffes qui saisissent l'objet physiquement par impact direct.
(b) Préhenseurs pénétratifs (ingressive gripper) : outils comme les épingles ou les aiguilles qui pénètrent la surface de l'objet pour le saisir (utilisés notamment dans la manipulation des fibres textiles, de carbone et de verre).
(c) Préhenseurs attractifs (astrictive gripper) : systèmes qui utilisent des forces d'attraction appliquées à la surface de l'objet, comme le vide, les forces magnétiques ou l'électroradhésion.
(d) Préhenseurs contigus (contigutive gripper) : mécanismes nécessitant un contact direct pour que l'adhésion se produise, par exemple à travers la colle, la tension superficielle ou la congélation. Figure adaptée de [117].

Les effecteurs, ou préhenseurs, sont des dispositifs utilisés pour saisir et manipuler des objets. Le choix de l'effecteur approprié dépend des objets à manipuler et des contraintes spécifiques de la plateforme utilisée. Les préhenseurs varient en termes de complexité, de mécanismes de fonctionnement et d'adaptabilité à différentes tâches.

Par exemple, la main Allegro (voir Figure 2.1) est un préhenseur hautement sophistiqué qui possède quatre doigts, chacun doté de trois articulations. Cette conception permet une manipulation fine et polyvalente des objets. Cependant, dans la littérature scientifique, la majorité des recherches se concentre sur les préhenseurs à pinces parallèles (à deux doigts), qui représentent l'un des cas les plus simples et les plus courants.

Les pinces parallèles sont appréciées pour leur simplicité de conception, leur facilité d'utilisation et leur capacité à saisir efficacement une large gamme d'objets. Cependant, elles peuvent être limitées dans leur capacité à manipuler des objets de formes complexes

ou à réaliser des tâches nécessitant une dextérité accrue.

Il est important de noter qu'il existe également de nombreux autres types d'effecteurs, tels que les préhenseurs à ventouses, les pinces à trois doigts, les pinces à rouleaux, les pinces magnétiques et les pinces à crochets, pour n'en nommer que quelques-uns. Chacun de ces effecteurs présente des avantages et des inconvénients spécifiques en fonction de l'application et des contraintes du système.

En somme, le choix d'un effecteur adapté est crucial pour la réussite de la manipulation d'objets par des robots. Une compréhension approfondie des différentes options disponibles et de leurs performances dans diverses situations permettra aux chercheurs et aux ingénieurs de concevoir des systèmes robotiques de manipulation d'objets plus efficaces et polyvalents.

2.1.2 Variété d'objets à saisir

La diversité des objets utilisés dans la recherche en saisie robotique est vaste [71]. Les premières études sur la synthèse de saisie analytique se sont concentrées sur la manipulation de formes simples, telles que des polygones (2D) et des polyèdres (3D) [130, 110].

Des objets du quotidien sont également couramment employés, comme ceux présents dans la base de données d'objets et de modèles YCB (Yale-CMU-Berkeley) [27] (Fig. 2.2). Cette base comprend les modèles 3D d'un grand nombre d'objets, ainsi que des objets physiques réels, tels que des couverts, des jouets et des outils de différentes formes, tailles, textures, poids et rigidités.

(a) Les objets de type aliments de la base de données YCB

(b) Les objets de type cuisine de la base de données YCB

(c) Les objets de type outils de la base de données YCB

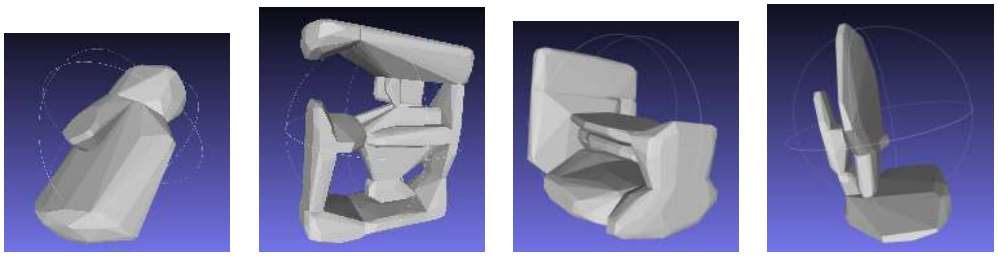
(d) Les objets de type formes de la base de données YCB

FIGURE 2.2 – Objets de la base de données YCB : (a) aliments, (b) cuisine, (c) outils, et (d) formes.

Des bases de données plus spécifiques peuvent également être utilisées. Par exemple, [172] a créé une base de données de 217 objets appartenant à 10 catégories courantes dans les dépôts de déchets radioactifs : bouteilles en plastique, canettes, chaînes, chiffons de nettoyage, gants, objets métalliques, tuyaux en plastique, raccords de tuyaux, éponges et blocs de bois.

Lors du défi Amazon Picking [41], les participants devaient manipuler 25 objets couramment vendus en ligne. La compétition a souligné les difficultés pratiques rencontrées par les robots face à des objets déformables, des surfaces transparentes ou réfléchissantes, etc.

Peu de travaux se concentrent sur des objets articulés, flexibles ou déformables [156], ou difficiles à percevoir, comme les objets transparents [155, 72, 197, 72]. [72] propose une méthode pour extraire une carte de profondeur de haute qualité à l'aide d'un réseau de neurones Neural Radiance Fields (NeRF), permettant ainsi de récupérer la géométrie d'objets transparents pour les manipuler.



(a) (b) (c) (d)

FIGURE 2.3 – Exemples d'objets générés par AGOD-Grasp [2].

Enfin, certains travaux [184, 118, 2] ont généré des objets spécialement conçus pour la préhension robotique, imprimables en 3D ou utilisables en simulation. L'Evolved Grasping Analysis Dataset (EGAD) [118] utilise des Compositional Pattern Producing Networks (CPPNs) pour créer des objets dotés d'une symétrie radiale et emploie l'algorithme évolutionnaire MAP-Elites [121] afin d'assurer une diversité géométrique et une couverture uniforme de l'espace 2D, en termes de difficulté de préhension et de complexité de forme. De manière similaire à EGAD, AGOD-Grasp [2] utilise un Variational AutoEncoder (VAE) 3D qui traite des grilles de voxels de taille 64×64×64 pour générer des objets sans symétrie radiale (Fig. 2.3). Ces objets, ainsi créés, garantissent une variété de formes et de difficultés de saisie, les rendant utiles pour l'entraînement et l'évaluation des systèmes de préhension robotique.

Le Tableau 2.1 contient une brève description de chaque ensemble de données dans l'ordre chronologique de leur publication, tandis que le Tableau 2.2 les analyse selon nos critères d'intérêt mentionnés ci-dessus.

TABLE 2.1 – Ensembles de données d'objets appropriés à des fins de préhension, extrait de [2].

Dataset	Description	Format	Year
MATAMOROS et al. [112]	Tableware, cutlery, trash bags, bags, disks, books, coat rack, trays, pourable (muesli, cereal), heavy objects, tiny objects, fragile objects, amorphous objects	Physical objects, handed to Robocup participants	2009-2019
CHOI et al. [37]	Medical items, dining items, bathroom items, living room items, bed room items, personal belongings	Shopping list (outdated)	2009
JIANG et al. [75]	1035 images of 280 different objects (different orientations, associated point cloud, background image)	The raw dataset consists of : (a) images, (b) grasping rectangles, (c) pointclouds, (d) background images and (e) a file containing a mapping from each image to the corresponding background image.	2011
KASPER et al. [81]	Boxes, cans, tubes, mugs, bottles, toys, pliers. Limited variation in object types. 3D models are reconstructions, not sources.	3D triangulated mesh, 2D image data, texture information, grasp information.	2012
ÇALLI et al. [29]	Food items, kitchen items, tools, shape imate,s task items (toys, magazine)	Point clouds, associated meshes and texture, mass, dimensions, models for integration into planning and simulation software. Meshes reconstructed from scanning (without bottom side of objects).	2015
CORRELL et al. [41]	25 products commonly sold on Amazon, with various shapes, sizes, deformable shapes, transparent and reflective surfaces, fragile.	Shopping list of 25 objects commonly sold on Amazon.	2018
Google [46]	Procedurally generated dataset	3D models of 1000 objects in Wavefront OBJ format.	
MORRISON et al. [118]	Objects automatically generated using CPPN	2000 objects in total, of which 49 objects are printable without supporting structures	2020
ANDRIES et al. [2]	Objects automatically generated in voxelgrid format using a VAE, then converted to meshes.	50 objects selected out of several hundred thousand objects.	2023

TABLE 2.2 – Ensembles de données d'objets analysés selon des critères d'intérêt extrait de [2].

Dataset	Object variability	RGB-D scans	Simulation models	Physical models	3D printable	Physically deliverable
[112]	✓	✗	✗	✓	✗	✓
[37]	✓	✗	✗	✗	✗	✗
[75]	✓	✓	✗	✗	✗	✗
[81]	✗	✓	✓	✗	✗	✗
[29]	✓	✓	✓	✓	✗	✓
[41]	✓	✗	✗	✓	✗	✓
[46]	✗	✗	✓	✗	✓	✗
[118]	✓	✗	✓	✓	✓	✗
[2]	✓	✗	✓	✓	✓	✗

2.1.3 Préhension d'objets isolés et de groupes d'objets

La majorité des méthodes de synthèse de préhension se concentrent sur des objets isolés reposant sur des surfaces planes. Les scènes comportant plusieurs objets présentent des difficultés supplémentaires : l'espace de travail accessible autour des objets à saisir est fortement limité, les occlusions sont plus susceptibles d'entraver la perception des objets, et identifier la saisie adéquate peut être plus complexe en raison des interactions de contact entre les objets lors de l'action de saisie.

Pour les méthodes capables d'appréhender des scènes avec plusieurs objets, on distingue également différents niveaux de difficultés, comme traiter des objets plutôt petits et légers répartis dans un bac, les tas structurés, une configuration compacte d'objets principalement plus grands et plus lourds (placards de cuisine, rayons de supermarché), et les tas denses (Fig. 2.4).

FIGURE 2.4 – Trois exemples de scènes selon les niveaux de difficulté : un objet isolé, de petits objets légers répartis dans un bac, et un tas dense.

Enfin, certaines approches dites "target-driven" visent à identifier et saisir un objet spécifique parmi un tas [103, 100, 103]. Ces méthodes se concentrent sur la reconnaissance et la localisation de l'objet cible, tout en tenant compte des contraintes et obstacles présents dans l'environnement pour réussir la saisie de manière optimale.

2.1.4 Prise d'objet dans les espaces SE(2) et SE(3)

La préhension dans les espaces SE(2) ou 4DOF (Degrees Of Freedom, soit degrés de liberté en français) implique que l'objet cible se situe dans un environnement plan. La préhension est limitée à une direction unique dans ce contexte. Ici, SE(2) fait référence à l'ensemble des transformations rigides dans un espace bidimensionnel, comprenant des mouvements de translation le long de deux axes et une rotation. Ainsi, la représentation de la préhension est simplifiée, passant de 6 dimensions (voir Sec. 2.2) à 3, incluant au minimum les positions 2D sur le plan et l'angle de rotation en une dimension. Nous identifions deux types de méthodes pour évaluer cette préhension : celles qui évaluent les points de contact et celles qui évaluent les rectangles orientés.

En contraste, la préhension dans l'espace SE(3) ou 6DOF permet au dispositif de préhension de manipuler l'objet sous différents angles dans l'espace tridimensionnel, sans simplification de la posture du dispositif en 6 dimensions. SE(3) représente l'ensemble des transformations rigides dans un espace tridimensionnel, qui comprend des mouvements de translation le long de trois axes et des rotations autour de ces axes. Les méthodes SE(3) peuvent être divisées en deux catégories, selon qu'elles utilisent un nuage de points à vue unique ou un modèle complet de l'objet : les méthodes basées sur le nuage de points partiel et les méthodes basées sur le modèle complet. Pour ces dernières, les préhensions peuvent être prédéterminées manuellement ou par simulation, transformant ainsi le problème en une question d'estimation de la pose d'un objet en 6 dimensions.

2.1.5 Méthodes de préhension sans modèle et basées sur un modèle

La synthèse de préhension, qui est le processus de détermination des points de préhension optimaux pour un robot, peut être approchée via deux catégories principales de méthodes : les méthodes basées sur un modèle complet (qui nécessitent l'accès à un modèle 3D complet ou à un modèle scanné de l'objet [12]), et les méthodes basées sur un nuage de points partiel de l'objet.

Il est à noter que ces approches sont principalement valides pour des objets dont les modèles étaient déjà connus. Pour des objets inédits ou atypiques (par exemple, un morceau de brique cassé de manière aléatoire, une canette comprimée de manière non uniforme, un objet brisé de manière imprévue), l'approche basée sur le modèle peut s'avérer inapplicable, ou nécessiter une étape préliminaire de reconstitution de la forme de l'objet à partir de plusieurs prises de vue ou à partir de l'expérience accumulée par le système.

Les méthodes basées sur le modèle complet s'appuient sur une gamme de techniques, notamment l'estimation de la pose de l'objet en 6D et les méthodes de complétion du modèle. On peut identifier plusieurs sous-catégories parmi ces approches :

- La localisation des objets sans classification, qui fournit uniquement les régions potentielles des objets ciblés sans préciser leur catégorie.
- La détection des objets, qui fournit non seulement les boîtes englobantes des objets ciblés, mais aussi leur catégorisation.
- La segmentation par instance de l'objet, qui divise les régions de pixels des objets ciblés en fonction de leur catégorie, permettant ainsi une identification plus précise.

D'autre part, les méthodes basées sur le nuage de points partiel exploitent des techniques d'estimation des prises candidates et des méthodes de transfert des prises à partir d'une base de données de prises existantes. Ces approches permettent de déterminer les

saisies possibles en se basant sur des informations partielles de l'objet, éliminant ainsi le besoin d'un modèle 3D complet. Cette flexibilité offre un avantage certain lors de la manipulation d'objets non familiers ou de formes complexes.

2.1.6 Méthodes de préhension en boucle ouverte et en boucle fermée

La majorité des algorithmes de saisie nécessitent de longs temps d'exécution et fonctionnent donc en boucle ouverte (sans rétroaction). Cela peut entraîner des échecs si la position d'un objet change entre la perception de la scène et l'exécution de la saisie.

FIGURE 2.5 – KALASHNIKOV et al. [79] utilisent des exemples réels (580K tentatives de saisie) pour entraîner un Markov Decision Process (MDP) model.

À l'inverse, une commande en boucle fermée, comme celle proposée dans [79, 201, 179, 33, 98, 169, 119, 120], met continuellement à jour la stratégie de saisie du robot en fonction des observations les plus récentes afin d'optimiser le succès de la saisie à long terme (Fig. 2.5).

Ces politiques en boucle fermée peuvent présenter des comportements correctifs, des mouvements de sondage pour déterminer la meilleure saisie, un repositionnement non préhensile des objets et d'autres caractéristiques qui ne sont réalisables que lorsque la saisie est formulée comme un processus dynamique en boucle fermée.

Cependant, ces politiques ne formulent souvent pas explicitement la saisie à effectuer, ce qui rend difficile l'intégration avec d'autres modules robotiques (tels que la planification de trajectoires et l'évitement d'obstacles) ou la spécification de tâches de plus haut niveau, même après ré-entraînement (par exemple, la sélection d'objets spécifiques dans un tas).

2.1.7 Actions préliminaires à la manipulation

La saisie robotique est généralement composée de plusieurs étapes : la détection de l'objet, la planification de la trajectoire, la préhension, et enfin, le levage et le repositionnement. En plus de certaines approches en boucle fermée mentionnées précédemment, plusieurs travaux intègrent également des actions non préhensiles, par exemple pour déplacer les objets avant de les saisir [132, 14, 50]. Ainsi, Zeng et al. [204] ont proposé un cadre d'apprentissage par renforcement Q-learning pour combiner la préhension et la poussée d'objets.

D'autres approches prometteuses [194, 180, 48, 105] consistent à exploiter les modèles de prédiction visuelle [53] pour informer une méthode de commande prédictive modèle (MPC) visuelle. Le principe de ces approches repose sur la prédiction de la sortie visuelle de la caméra du robot après l'exécution d'une action ou d'une séquence d'actions, et l'utilisation de cette prédiction pour orienter un contrôleur optimal ou un planificateur de décisions. Cette stratégie permet d'anticiper les conséquences des actions et d'ajuster les mouvements du robot en conséquence, améliorant ainsi les performances de manipulation.

2.2 Représentations des configurations de saisie

Il existe plusieurs types de représentation de préhension.

2.2.1 Représentations de saisie basées sur des points

Un ensemble de points de contact exprimés dans le repère de l'objet constitue cette représentation. Elle est adaptable à différentes pinces avec différents nombres de doigts et est largement utilisée dans les méthodes analytiques (Sec. 2.4) ou lors de l'utilisation d'effecteurs avec de nombreux doigts (Fig. 2.1).

[159] utilisent un seul point 3D dans leur synthèse de préhension sur des observations multivues d'images RGB. Réduite à un seul point, cette représentation est largement utilisée pour les prises par succion [107, 31, 74].

2.2.2 Régions de contact indépendantes (Independent Contact Regions)

Plus complexes à calculer que la représentation basée sur le contact, les Independent Contact Regions (ICRs) [129] définissent un ensemble de régions indépendantes sur la surface de l'objet, de sorte que placer un doigt sur chaque ICR entraîne une prise en *force closure* (les forces de contact permettent de maintenir l'équilibre face à n'importe quelles forces ou moments extérieurs appliqués sur l'objet), offrant ainsi des prises plus robustes face aux imprécisions d'exécution (voir Sec. 2.4).

2.2.3 Préhensions dans l'espace SE(3)

Cette représentation simplifiée profite du fait que les points de contact sur un objet sont entièrement déterminés par la pose 6D du préhenseur $g = (x, y, z, r_x, r_y, r_z)$ dans le cas des préhenseurs à pinces parallèles majoritairement utilisés dans la littérature scientifique. Elle prend en compte la position en 3D (x, y, z) et l'orientation 3D (r_x, r_y, r_z) du préhenseur.

2.2.4 Représentation de saisie par rectangle orienté

Cette représentation projette le point central de l'outil (TCP) et l'orientation de la préhension dans le plan de l'image et la définit comme un rectangle orienté [89, 119, 49, 97]. Elle est souvent utilisée pour comparer les performances des méthodes de préhension 4DOF (Sec. 2.1.4).

Présentée dans [106], elle est caractérisée par la position centrale de l'extrémité de l'outil de préhension (x, y), l'orientation $\theta \in [-\frac{\pi}{2}, +\frac{\pi}{2}]$, la longueur de l'ouverture de la pince l, et la largeur du rectangle (toutes deux en pixels) : $(x, y, \theta, l, width)$ (Fig. 2.6).

FIGURE 2.6 – (a) La représentation par JIANG et al. [75] : Sommet supérieur (r_G, c_G), longueur m_G, largeur n_G et son angle par rapport à l'axe des x, θ_G pour un ustensile de cuisine. Il peut y avoir plusieurs prises définies comme indiqué. (b) La représentation simplifiée par REDMON et al. [147] pour un marteau, montrant son centre de préhension à (x, y) orienté selon un angle de θ à partir de son axe horizontal. Le rectangle a une largeur et une hauteur de w et h respectivement. Image provenant de [24].

2.2.5 Cartes de saisie au niveau des pixels (Pixel-level grasp maps)

Proposées par [4] et inspirées par les travaux sur la segmentation d'objet, ces modèles prennent en entrée des images (RGB et/ou D) et fournissent en sortie une image segmentée de la qualité de préhension à chaque pixel et éventuellement des paramètres de préhension associés (largeur, hauteur et orientation).

2.3 Utilisation de capteurs pour manipulation robotique

L'aptitude à manipuler des objets repose fortement sur la compétence des robots à percevoir et à analyser l'environnement qui les entoure. Pour accomplir cette mission essentielle, ils exploitent un éventail de capteurs spécialisés, chargés de recueillir des données pertinentes concernant l'environnement et les objets qui s'y trouvent. Les capteurs visuels et tactiles, deux catégories majeures, jouent un rôle clé dans ce processus.

2.3.1 Capteurs visuels

Les capteurs visuels, tels que les caméras RGB-D, sont un type de capteur clé utilisé en robotique pour des tâches telles que la reconnaissance, la localisation et la saisie d'objets. Les caméras RGB-D fournissent à la fois des informations de couleur et de profondeur, permettant aux robots de percevoir et de comprendre leur environnement de manière plus détaillée et nuancée.

2.3.1.1 Représentations 3D

L'utilisation des capteurs de profondeur RGB-D abordables, tels que la Kinect de Microsoft, introduits dans les années 2010, a stimulé de nombreux travaux en robotique, notamment dans le domaine de la synthèse de préhensions. Les approches varient en fonction des données exploitées, notamment les images RGB-D [106, 96, 147] (voir Fig. A.4), les nuages de points [197, 122, 144, 173], les octrees [5] et les fonctions de distance signée tronquée (TSDF) [19, 169] (voir Fig. 2.7).

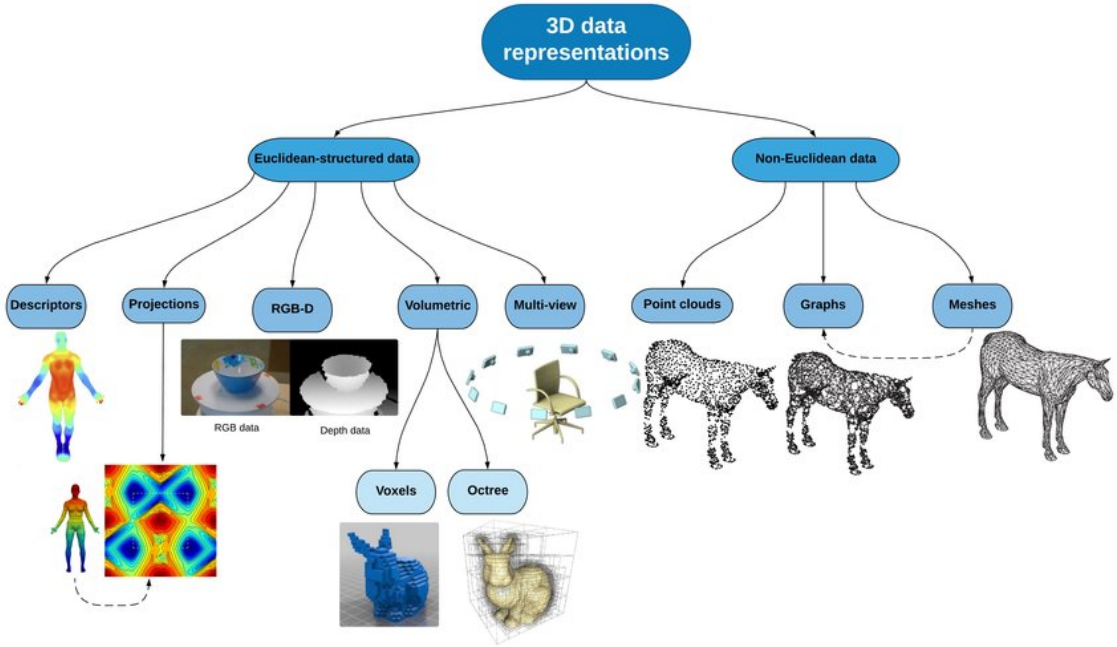


FIGURE 2.7 – Diverses représentations pour les données 3D : Représentation euclidienne (Descripteurs [87], projections [111], RGB-D [91], volumétrique ; voxels et octree [95] et multi-vues [171]) et représentations non euclidiennes (nuages de points, graphes et maillages). Figure adaptée de [1]

Les informations 3D peuvent être représentées de multiples façons, comme le montre la Fig. 2.7.

L'apprentissage profond sur les données de nuages de points 3D a commencé bien plus tard que son succès fulgurant sur les images RGB. Au début, les données 3D étaient représentées sous forme de voxels 3D [114] ou par l'extraction de caractéristiques à partir d'images de profondeur 2,5D [61] et leur traitement similaire à celui des images RGB à

l'aide de réseaux de neurones convolutifs. Cependant, ces méthodes n'apportaient souvent que des améliorations marginales. Qi et al. [142, 143] ont introduit une nouvelle architecture, appelée PointNet et PointNet++, capable de représenter les données 3D et d'extraire efficacement des représentations. Le succès de PointNet a conduit à l'introduction de différentes variantes d'architectures de réseaux [170, 189] pour représenter les données 3D, montrant une amélioration significative de l'estimation de la pose des objets 3D, de la segmentation sémantique et de la segmentation des pièces [170, 143, 141, 198].

2.3.2 Capteurs tactiles

Les capteurs tactiles jouent un rôle essentiel dans la robotique en permettant aux robots d'interagir physiquement avec leur environnement et de détecter quand un objet est en contact avec leur pince ou manipulateur. Ces capteurs peuvent fournir des informations précieuses sur la forme, la taille, la texture, la position et l'orientation d'un objet par rapport au robot [52, 7]. Ils sont essentiels pour les tâches de saisie et de manipulation d'objets, permettant aux robots de s'adapter à diverses situations et d'améliorer leur performance.

2.3.2.1 Types et matériaux des capteurs tactiles

Les capteurs tactiles peuvent être classés en différentes catégories selon leur principe de fonctionnement, telles que les capteurs capacitifs (constitués d'un condensateur, dans lequel la distance entre les plaques ou la surface effective est modifiée lorsqu'une force est appliquée), optiques (la trajectoire ou l'intensité de la lumière d'une LED est modifié par le toucher est mesurée par un photodétecteur), piézorésistifs (fonctionnent sur le principe selon lequel un métal conducteur change de résistance lorsqu'il est sous pression), piézoélectriques (produisent une charge électrique proportionnelle à la force appliquée), inductifs, optoélectroniques et magnétiques [52]. Ces capteurs peuvent être fabriqués à partir de divers matériaux, des matériaux actifs ou des électrodes flexibles, ce qui les rend adaptés à diverses applications, de la robotique à la santé et même à la chirurgie [182].

2.3.2.2 Capteurs tactiles basés sur la vision

Parmi les capteurs tactiles, les capteurs basés sur la vision, tels que le DIGIT [92], sont particulièrement intéressants pour les méthodes d'apprentissage basées sur les données, en particulier celles qui exploitent les techniques d'apprentissage profond pour le traitement des images. Le DIGIT est un capteur tactile qui utilise un élastomère déformable, comme le GelSight [203], pour mesurer les changements de surface causés par la force de contact. Ces capteurs peuvent être montés sur des préhenseurs existants et sont adaptés pour manipuler et saisir avec précision des objets, y compris des objets mous [166].

Le développement de ces capteurs tactiles basés sur la vision est soutenu par des outils de simulation tels que TACTO [187], un simulateur pour les capteurs tactiles basés sur la vision, qui génère des lectures tactiles haute définition et peut être configuré pour simuler des capteurs comme DIGIT. Cela facilite le développement et l'évaluation des performances de ces capteurs dans diverses applications robotiques.

2.3.2.3 Intégration des capteurs tactiles dans la manipulation robotique

L'intégration de capteurs tactiles basés sur la vision dans la manipulation robotique, et plus particulièrement dans la préhension, peut améliorer les performances en fournissant des informations supplémentaires sur l'emplacement du contact, les forces de contact, la forme de l'objet, les conditions de contact et la texture [7]. Les études menées avec le capteur DIGIT [23, 22] ont montré que l'utilisation de réseaux neuronaux profonds pour la prédiction des résultats de saisie basée sur la vision, le toucher et la combinaison de la vision et du toucher peut améliorer la réussite de la préhension et la manipulation d'objets.

Dans une application spécifique, le capteur DIGIT a été utilisé pour manipuler avec précision des objets tels que des câbles d'écouteurs [166]. Un serre-câble en circuit fermé s'appuie sur le retour tactile en temps réel du capteur pour saisir un câble d'écouteur, faire glisser les doigts vers le connecteur jack et l'insérer avec succès.

Cependant, la performance opérationnelle des capteurs tactiles basés sur la vision n'est pas entièrement connue. Il est important d'évaluer si la mesure optique du capteur est corrélée avec la force et le couple de contact, car ces informations sont cruciales pour évaluer la stabilité et le succès de la préhension à l'aide des mesures traditionnelles basées sur la force. Ainsi, des études expérimentales, telles que celle présentée dans le chapitre 4, sont nécessaires pour caractériser et évaluer les performances de préhension des capteurs tactiles tels que le DIGIT.

En somme, les capteurs tactiles, en particulier ceux basés sur la vision, offrent de nouvelles opportunités pour améliorer les performances de saisie et de manipulation d'objets dans la robotique. Les recherches et développements continus dans ce domaine contribuent à l'évolution des capteurs tactiles et à leur intégration réussie dans diverses applications robotiques.

2.4 Approches géométriques de la préhension

Avec la connaissance partielle des propriétés physiques de l'objet manipulé (masse, forme, centre de masse, coefficient de friction), plusieurs méthodes ont été proposées pour générer des candidats de préhension [86]. Ces méthodes sont dénommées géométriques ou également appelées analytiques.

L'objectif de la synthèse est de satisfaire les propriétés de form-closure (la géométrie de la prise contraint complètement l'objet) et de force-closure (les forces de contact permettent de garder l'équilibre face à n'importe quelles forces ou moments extérieurs appliqués sur l'objet).

Diverses métriques de qualité ont été proposées pour évaluer les préhensions, telles que [55, 152, 148]. Parmi celles-ci, la métrique ϵ de Ferrari Canny [55] a démontré sa capacité à prédire le succès de la saisie. Cette métrique évalue la capacité d'une préhension à résister à des perturbations de force externes dans n'importe quelle direction, sans glissement. Elle est calculée en utilisant le rayon du plus petit ballon pouvant être inscrit dans l'espace des forces de préhension possibles. Une valeur plus grande pour la métrique ϵ indique une meilleure préhension, car elle signifie que la préhension peut résister à des forces plus grandes avant de perdre le contrôle de l'objet.

Les préhensions sont généralement représentées par des points de contact ou par des

régions de saisie stables (Independent Contact Regions, ICRs) définies par Nguyen [130] (voir Sec. 2.2). Les ICRs sont un ensemble de régions sur l'objet où chaque doigt peut être placé indépendamment, n'importe où, sans que la prise perde la propriété de force-closure.

D'autres approches consistent à rechercher des cages de préhension, qui sont des configurations de cages à partir desquelles un objet peut être saisi sans briser la cage au préalable [149]. Ces cages de préhension offrent une plus grande flexibilité et tolérance aux erreurs lors de la manipulation d'objets complexes ou dans des environnements incertains.

L'approche géométrique offre des solutions robustes pour la génération de préhension, en tenant compte des contraintes physiques et mécaniques de la manipulation. Toutefois, elle repose sur la connaissance préalable des objets et des propriétés environnementales, ce qui peut limiter son applicabilité dans des situations où ces informations sont incomplètes ou inexactes. Pour surmonter ces défis, des approches basées sur l'apprentissage automatique et la vision par ordinateur ont été développées, offrant des solutions plus adaptatives et flexibles pour la génération de préhension dans des environnements réels et complexes.

2.5 Approches basées par les données

Comme dans de nombreux domaines, les approches guidées par les données (également appelées empiriques) basées sur l'apprentissage automatique ont entraîné de nombreuses avancées pour la synthèse de préhension. Cela est dû à la plus grande disponibilité des données, à l'amélioration des algorithmes et à l'augmentation de la puissance de calcul des processeurs [11].

Les approches analytiques nécessitent généralement l'intervention d'experts, la compréhension de l'environnement et les capacités du robot. Les propriétés physiques de l'objet requises ne sont connues qu'avec des scénarios de fabrication bien contrôlés, car ces informations sont difficiles à obtenir avec les capteurs classiques des robots (capteurs visuels ou tactiles) [162]. De plus, ces méthodes ont tendance à être coûteuses en calcul pour des objets autres que simples, ce qui pose des problèmes pour les objets du quotidien [154].

D'où la popularité des travaux qui utilisent des approches plus axées sur les données [11] ou des approches combinant des techniques à la fois analytiques et basées sur les données [108, 106]. Ces approches empiriques tirent parti de l'apprentissage profond et des techniques de vision par ordinateur pour traiter des scénarios plus complexes et incertains, avec une plus grande adaptabilité et flexibilité.

Les méthodes empiriques peuvent utiliser des bases de données de préhension annotées, des simulations robotiques ou des apprentissages par renforcement pour entraîner des modèles capables de prédire les préhensions réussies pour une grande variété d'objets et de situations. Ces approches permettent également d'incorporer des informations multimodales, telles que les images RGB-D, les nuages de points 3D et les données tactiles, pour améliorer la précision et la robustesse des prédictions de préhension.

En résumé, les approches empiriques offrent des solutions prometteuses pour la génération de préhension dans des environnements réels et complexes, en tirant parti de l'apprentissage automatique et des techniques de vision par ordinateur pour surmonter les défis posés par les approches analytiques traditionnelles.

Dataset	number of objetcs	format
RGB-D Object Dataset [91]	51	207920 RGB-D
CORSMAL Containers [196]	23 object	1656 RGB-D
ModelNet [195]	127 915 3Dmesh	127 915 3D mesh
Johns et al. [77] subset of [195]	1000 objects	1000000 depth image
3DNet [193]	200 object classes	over 5150 3D models and over 3200 RGB-D images
Viereck et al. [179] subset of [193]	381 objects	500 000 depth images
RoboNet [45]	7 different robots	15 million video frames
ShapeNet [32]	55 object categories	51300 3D models
CMU dataset [138]	over 150	50567 RGB-D
BigBird [168]	125	600 RGB-D
Pas et al. [136] subset of [168]	55	216k grasps
Grasping Dataset [205]	over 60 Objects	1837 RGB-D
StanfordGrasping [158]	10	13747 RGB-D
Cornell Grasping [75]	240	885 RGB-D
Dex-Net 2.0 [106]	over 150	6.7 M(Depth only)
Jacquard [49] subset of [32]	11619	54485 RGB-D
Nuclear Waste Dataset [172]	217 - 10 categories	RGB-D video clips
multiObject [40]	3-5 different objects on each scene 35 objects in total. ,	96 RGB-D
LUV [174]	3640 fabric images, 486 cable images, 1364 needle images	RGB images
T-LESS [69]	30 industry-relevant objects	49000 RGB-D images
Siléane Dataset [17] subset of [69]	between 0 and 11 objects per scene	2601 RGB-D images
TaskGrasp [124]	250K task-oriented grasps , for 56 tasks and 191 objects	RGB-D information
SG14000 Dataset [101]	44 objects, 14k semantic grasps 7 tasks,6 object states	RGB-D
ContactDB [16]	50 objetcs	3750 meshes, 375K RGB-D-thermal frames
ACRONYM [54], meshes from [32]	8 872 objets	8 872 object meshes, 17,7 millions grasps

Table 2.3 – Bases de données utilisées pour l'apprentissage et l'évaluation des algorithmes de synthèse de saisies.

2.5.1 Bases de données dans la préhension robotique

[11] classifie les approches guidées par les données selon l'origine des données d'entraînement utilisées : acquises par essais et erreurs, étiquetage hors ligne ou sous forme de démonstration. On peut également ajouter la distinction entre les données issues de simulations et celles issues du monde réel.

2.5.1.1 Collectées de manière autonome et étiquetées au préalable

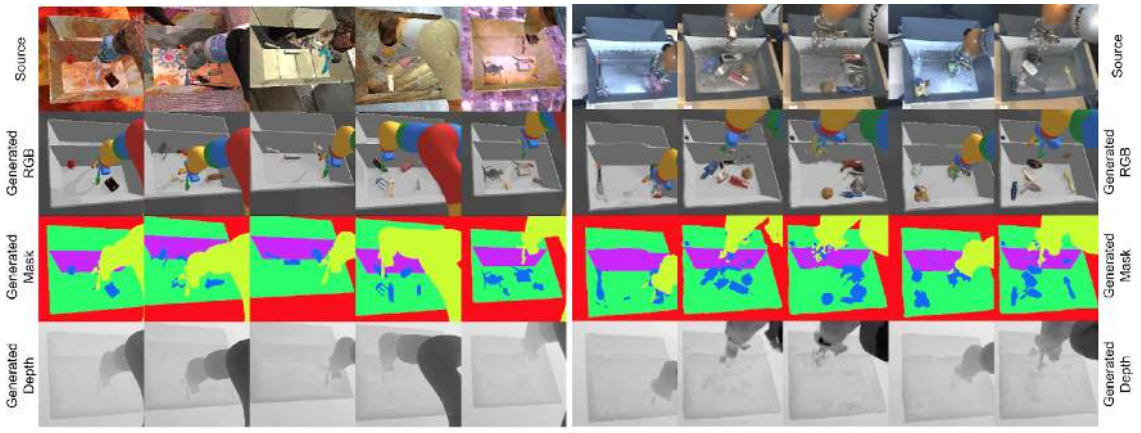
Les données du monde réel peuvent être récupérées par une interaction autonome avec l'environnement [45]. La collecte automatique de suffisamment de données d'entraînement par essais et erreurs avec des robots réels [138, 98] est cependant chronophage et ne fournit pas nécessairement de meilleurs résultats (par exemple, 50 000 essais et 700 heures d'utilisation du robot dans [138, 98]). Les bases de données étiquetées manuellement [75, 40] sont également souvent utilisées, la plupart présentent des objets uniques pour chaque

scène, [40] présentant 3 à 5 objets par scènes.

[174] Labels from UltraViolet (LUV) propose la collecte rapide de données étiquetées dans des environnements de manipulation réels sans étiquetage humain. LUV utilise une peinture transparente fluorescente aux ultraviolets avec des LED ultraviolettes programmables pour collecter des images appariées d'une scène sous un éclairage standard et sous un éclairage UV afin d'extraire de manière autonome des masques de segmentation et des points clés par segmentation des couleurs.

2.5.1.2 Issues de simulation

Une approche récemment couronnée de succès consiste à créer de grandes bases de données synthétiques à partir de modèles 3D et de simulations [77, 60, 106, 204, 122, 179, 49]. [73] ont montré que l'efficacité de cette nouvelle solution est comparable à celle de la solution entraînée uniquement sur des robots réels.



(a) Randomized-to-canonical samples. (b) Real-to-canonical samples.

FIGURE 2.8 – JAMES et al. [73] ont étendu le travail de KALASHNIKOV et al. [79] en remplaçant les données de manipulation préenregistrées par des données de simulation. Les données de simulation sont générées à l'aide de techniques de randomisation, puis adaptées au domaine réel.

DEPIERRE et al. [49] utilisent la représentation rectangulaire de [75] et génèrent des saisies positives et négatives dans un simulateur de physique.

MAHLER et al. [106] utilisent également un simulateur physique pour générer des prises sur des modèles d'objets 3D, qui peuvent être représentés comme une image de profondeur centrée sur le centre de la prise et alignée sur l'axe de rotation de la prise.

La force de ces bases de données est leur taille (des millions d'objets) (voir Tab. 2.3), mais elles ne prennent actuellement en compte que la forme 3D (via les données de profondeur) et supposent une distribution de masse uniforme des objets. Comme elles reposent fortement sur la simulation, elles ont souvent besoin de méthodes d'adaptation pour être efficaces avec des robots réels [15].

2.5.1.3 Démonstrations

Les démonstrations sont des exemples d'utilisateurs effectuant des actions. Elles sont souvent représentées sous forme de trajectoire définie dans l'espace des articulations, l'espace des tâches ou tout autre espace qui convient à l'expérience, les informations de forces sont parfois également utilisées.

Les développements récents représentent les mouvements avec des descripteurs invariants facilitant la réutilisation de la démonstration apprise dans de nouvelles situations ; par exemple, en s'appuyant sur la propriété de conditionnement gaussien comme dans l'approche ProMP (probabilistic movement primitive) [135], ou en encodant la démonstration dans de multiples systèmes de coordonnées [25]. De telles représentations utilisent les mêmes modèles pour reconnaître les actions existantes et en générer de nouvelles qui peuvent s'adapter à de nouvelles situations.

Classiquement, il y a trois façons de fournir ces démonstrations :

— **Les manipulations à main** : Les manipulations à la main sont naturelles, mais récupérer les données des démonstrations peut être difficile. Des gants peuvent être utilisés, ou les doigts peuvent être détectés dans une vidéo, mais cela peut être difficile. [64] facilite la détection des doigts en utilisant un ruban adhésif coloré, ce qui facilite la segmentation de ceux-ci (voir Sec. A.6). Avec ce genre de manipulation, il n'y a pas de garantie que la démonstration sera réplicable par le robot.
— **Le guidage kinesthésique** : Le guidage kinesthésique consiste à déplacer manuellement le robot. Comme les robots ont généralement accès à leurs configurations, les données de la démonstration sont directement récupérables et la trajectoire est exécutable par le robot. Cependant, la qualité des démonstrations peut être impactée par l'expérience des utilisateurs avec la manière particulière dont les robots se déplacent dans l'espace (par exemple, dans le cas de bras articulé possédant 7 DOF).
— **La téléopération** : La téléopération permet de commander à distance un robot en temps réel. Tout comme le guidage kinesthésique, les données de démonstration peuvent être récupérées directement et la trajectoire peut être reproduite par le robot. Divers systèmes de téléopération sont utilisés, pouvant inclure des retours visuels et/ou haptiques, et parfois tirer parti des technologies de réalité augmentée ou virtuelle. La littérature fait mention de divers dispositifs de téléopération, tels que les gants, les combinaisons de capture de mouvement, le suivi optique, la souris, le clavier, le joystick, l'interface utilisateur graphique (GUI) et les casques de réalité virtuelle/augmentée (VR/AR).

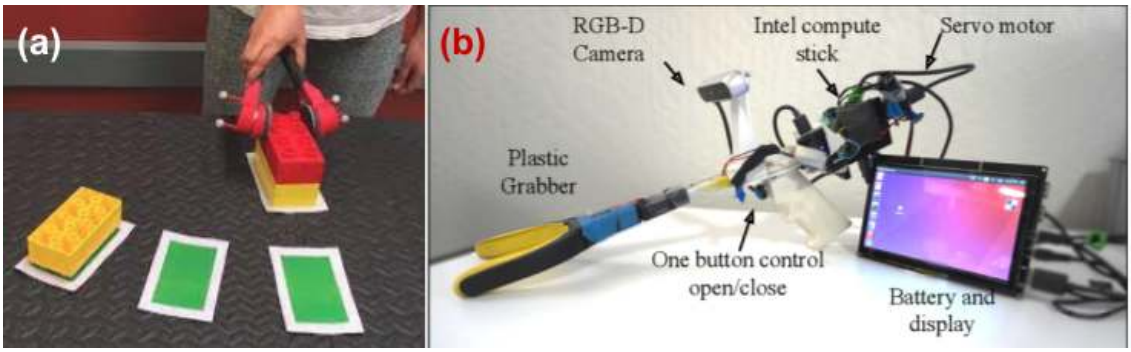


FIGURE 2.9 – (a) Utilisation de pinces instrumentées pour collecter une démonstration d'empilement de blocs [140], (b) dispositif portable de [169], consiste en une pince à deux doigts équipée d'une caméra RGB-D et d'un moteur qui contrôle l'ouverture binaire des doigts de la pince.

Pour la préhension, des appareils portatifs personnalisés conçus pour permettre de recueillir facilement des démonstrations dans divers environnements [140, 169] ont été également proposés (Fig. 2.9). Ces dispositifs permettent une collecte de données plus précise et adaptée à l'apprentissage du robot, tout en offrant un contrôle plus intuitif pour les utilisateurs. En utilisant ces méthodes de démonstration, les chercheurs et les ingénieurs peuvent développer des algorithmes d'apprentissage et des systèmes robotiques plus efficaces et adaptatifs pour la préhension et la manipulation d'objets dans divers scénarios.

2.5.2 Techniques d'apprentissage en préhension robotique

La préhension robotique est une tâche complexe et essentielle qui nécessite la capacité de comprendre et de manipuler une grande variété d'objets dans divers environnements. Cela comprend la reconnaissance des objets, la sélection des points de préhension appropriés, l'adaptation à la forme et à la taille des objets, et l'exécution de mouvements précis et contrôlés pour saisir et manipuler ces objets. Pour acquérir ces compétences, les robots peuvent utiliser diverses techniques d'apprentissage, qui diffèrent par la manière dont ils exploitent les données disponibles et par la manière dont ils modélisent le problème de préhension.

2.5.2.1 Méthodes basées sur l'imitation

La préhension robotique par imitation est une technique où un robot apprend à manipuler des objets en reproduisant les actions d'un humain ou d'un autre démonstrateur. Cette approche nécessite que le robot observe et interprète les actions du démonstrateur, par exemple comment la main ou les doigts sont positionnés autour de l'objet et la séquence de mouvements utilisée pour effectuer la prise. Le robot peut ensuite adapter ces informations à son propre comportement de préhension, lui permettant d'imiter la méthode de saisie du démonstrateur. Cette méthode est particulièrement efficace pour l'apprentissage de compétences de préhension, car elle tire parti de l'expérience et de l'expertise du démonstrateur. Elle peut également aider les robots à apprendre à mani-

puler des objets inconnus ou inhabituels, en se basant sur les conseils du démonstrateur concernant la stratégie de préhension appropriée [26].

Dans le contexte de l'apprentissage à partir de démonstrations (LfD), le robot observe un expert humain effectuant une tâche pour acquérir des exemples pertinents 2.5.1.3. En utilisant une base de données de saisies stables et les objets correspondants, les techniques de préhension peuvent être transférées à des objets similaires par imitation [134]. Cela permet d'éviter les difficultés inhérentes à l'exploration dans le cadre de l'apprentissage par renforcement 2.5.2.3, tout en communiquant directement les préférences de l'expert pour accomplir la tâche, ce qui peut être complexe à formaliser (par exemple, déterminer ce qui constitue une trajectoire optimale).

La politique d'imitation peut être déterminée en fonction de la similarité entre les modèles cibles et les objets dans la base de données d'apprentissage, ou entre la configuration actuelle du robot et la prise donnée. Ainsi, il est souvent nécessaire de reconnaître le type de préhension effectué par un humain lorsqu'il saisit un objet afin de générer une saisie similaire [206]. Pour effectuer cette classification, une catégorie spécifique est attribuée à chaque préhension à partir d'une taxonomie prédéfinie, comme celle de Cutkosky [43]. La reconnaissance de la saisie peut être basée soit sur la posture finale de la main, soit sur la séquence complète de la saisie, y compris le mouvement du bras.

Le Behavior Cloning est une méthode simplifiée de l'apprentissage par imitation, où le problème est reformulé en apprentissage supervisé. Les démonstrations sont traitées comme des données étiquetées, et le modèle est formé pour associer les données d'entrée (observations) à une action (démonstration) [160]. Nair et al. [128] ont combiné l'apprentissage auto-supervisé (SSL) et le Behavior Cloning avec des images en entrée pour développer un modèle de dynamique inverse au niveau des pixels pour une tâche de manipulation robotique de corde.

Zhang et al. [208] ont recouru à un casque de réalité virtuelle et un contrôleur VR pour recueillir des démonstrations sous forme d'images RGB-D. Ces images ont ensuite été utilisées pour entraîner une politique profonde en utilisant un réseau neuronal convolutif (CNN). Cette approche a permis de créer une méthode d'apprentissage efficace et flexible pour la préhension robotique.

2.5.2.2 Méthodes basées sur l'échantillonnage et la classification des candidats

Certaines approches de synthèse de préhension échantillonnent une gamme de saisies potentielles, puis les classent selon une mesure de qualité. Cette méthode, qui découple l'échantillonnage de la saisie et la classification, est plus interprétable et scalable. Cependant, son efficacité et sa rapidité dépendent largement de l'échantillonneur de préhension. Les échantillonneurs peuvent être basés sur des heuristiques ou sur des techniques d'apprentissage automatique.

Chaque scène peut présenter une multitude de candidats, rendant l'évaluation exhaustive de tous les candidats infeasible. Les échantillonneurs adoptent diverses stratégies pour limiter le nombre de candidats à évaluer. Par exemple, Pinto et Gupta [138] évaluent les saisies uniquement par leur position, générant ensuite des scores pour 18 orientations possibles.

Lenz et al. [96] adoptent une détection en cascade à deux étapes pour les saisies en

4DOF (Sec. 2.1.4), où un modèle plus léger évalue rapidement de nombreux candidats, puis un modèle plus complexe évalue les candidats les plus prometteurs.

Mahler et al. [106] utilisent un seul modèle, mais au lieu d'échantillonner de manière exhaustive, ils emploient la méthode de l'entropie croisée (CEM) [153] pour localiser les saisies de haute qualité. La CEM procède par cycles itératifs d'échantillonnage et d'évaluation des candidats en utilisant le modèle. Après chaque cycle d'évaluation, un petit nombre de prises de haute qualité sont sélectionnées, puis un nouvel échantillonnage est effectué près de ces prises.

Certaines approches suggèrent de modéliser les objets à saisir en primitives de forme, comme des cuboïdes et des superquadriques [178, 137], avant de planifier la préhension. Bien que ces méthodes soient assez interprétables et prévisibles dans leur sortie, la simplicité de la représentation limite la gamme de formes qui peuvent être modélisées, affectant les performances de la planification de la préhension.

Pas et al. [136] détectent les saisies à 6 DoF sur des nuages de points, en supposant une pince à mâchoires parallèles. Le nuage de points de la scène est échantillonné uniformément, la direction d'approche étant fixée par la normale à la surface. Les candidats représentés dans le repère de Darboux (définis par les courbures principales d'une surface) sont filtrés en utilisant une série de critères géométriques.

Les échantillonneurs peuvent également être formés via l'apprentissage supervisé, ce qui les rend généralement plus performants. [199] Yang et al. propose un modèle qui prend une image RGB-D et un masque de segmentation en entrée et alimente deux réseaux neuronaux qui génèrent respectivement une prise 6-DOF et une reconstruction du nuage de points 3D de l'objet.

Mousavian et al. [122] et Murali et al. [125] utilisent un VAE (Variational Autoencoder) pour échantillonner plusieurs propositions de préhension, accélérant ainsi la génération de candidats. Le VAE prédit une saisie sur un nuage de points segmenté correspondant à un objet. Il comprend un encodeur, Q(z|X,g), et un décodeur, P(G|X,z), qui sont tous deux conditionnés par le nuage de points et entraînés à l'aide d'une fonction objectif VAE standard. L'encodeur et le décodeur sont implémentés en utilisant PointNet++ [143], et la pose de la pince est représentée par des points sur la surface de la pince qui sont ajoutés au nuage de points tout en étant identifiés comme des points de la pince. Après l'entraînement, les candidats peuvent être générés par échantillonnage à partir de la distribution normale standard et l'évaluation du décodeur P.

La fonction d'évaluation de ces candidats est généralement apprise. Les méthodes varient en fonction du modèle utilisé et de l'encodage des saisies. Lenz et al. [97], Pinto et al. [138], Mahler et al. [106] et Pas et al. [136] représentent les saisies sous forme d'image centrée sur la position de la saisie, tandis que Liang et al. [99] et Murali et al. [125] utilisent PointNet pour traiter directement les nuages de points.

L'apprentissage non supervisé

L'apprentissage non supervisé traite des données non étiquetées et cherche à découvrir les relations internes des échantillons, permettant de modéliser les densités de probabilité sur les entrées [185]. Il comprend principalement la réduction de la dimensionnalité et le clustering. La première cherche à simplifier les données en conservant les caractéristiques essentielles, tandis que le clustering vise à regrouper ou à segmenter les données en groupes en fonction de leur similarité.

Des travaux tels que [176, 202] utilisent les K-means comme méthode de clustering

pour segmenter le nuage de points en fonction de leurs caractéristiques, ce qui facilite la compréhension de la structure des données et rend la tâche d'apprentissage plus accessible pour le robot.

Contrairement aux approches traditionnelles d'apprentissage non supervisé, les Variational Autoencoders (VAE) et les Generative Adversarial Networks (GAN) sont des modèles génératifs. Ces modèles apprennent la distribution des données à partir de l'ensemble d'apprentissage, puis utilisent le modèle et la distribution apprise pour générer et modéliser de nouvelles données. En capturant la structure latente des données et en générant de nouvelles instances plausibles, ces modèles génératifs ont le potentiel d'apprendre de manière flexible et adaptative.

Certains travaux ont adopté cette stratégie dans l'étape de génération de candidats de préhension. Inspirés par les GAN, [122, 125] utilisent le VAE pour échantillonner plusieurs propositions de préhension, accélérant ainsi la génération de candidats. Ces approches exploitent la puissance des modèles génératifs et ont le potentiel d'améliorer l'efficacité et la robustesse des algorithmes de préhension robotique. Elles sont capables de s'adapter aux variations et aux complexités des scénarios de manipulation d'objets du monde réel, ce qui rend ces techniques particulièrement pertinentes pour le domaine de la robotique.

2.5.2.3 Apprentissage de bout en bout

L'émergence de l'apprentissage profond a permis l'intégration de la tâche de préhension directement au sein d'un réseau de neurones entraîné de bout en bout. Ces réseaux prennent en entrée des données brutes, par exemple, des informations issues de capteurs tactiles ou de caméras, et en sortie, ils produisent la configuration appropriée pour la saisie. Toutes les étapes intermédiaires, y compris l'échantillonnage des saisies et l'évaluation de leur qualité, sont régulées par les mises à jour du modèle lors de l'entraînement. Ces méthodes tendent à s'exécuter plus rapidement que les deux types d'approches précédemment mentionnés.

Redmon et al. [147] ainsi que Kumra et al. [90] ont reformulé la détection de préhension comme un problème de régression. Ils ont proposé des modèles basés respectivement sur les architectures AlexNet [88] et ResNet [63], qui prennent en entrée l'image de la scène entière et génèrent une saisie. Par nature, cette méthode ne peut fournir qu'un seul candidat par scène. Pour l'entraînement sur le Cornell dataset, une saisie positive est sélectionnée aléatoirement par scène.

Redmon et al. [147] ont par la suite étendu leur méthode en décomposant la scène en une grille 7×7, où chaque cellule est donnée en entrée à leur modèle, générant une saisie pour chaque cellule de la grille. Johns et al. [77] ont formulé le problème sous forme de classification binaire, en prédisant la probabilité qu'une saisie existe pour chaque centre d'une grille 45×34 et pour 6 orientations possibles.

Les réseaux de proposition de région (RPN), comme R-CNN, sont un type d'architecture de réseau neuronal fréquemment utilisé dans les tâches de détection d'objets. L'objectif principal d'un RPN est de générer un ensemble de boîtes englobantes qui sont susceptibles de contenir des objets d'intérêt dans une image d'entrée. Dans le contexte de la préhension robotique, il est possible d'utiliser des RPNs, mais en les formant à prédire des saisies, par exemple sous forme de rectangles orientés 2.2.4. Chu et al. [39] proposent une séquence de deux modèles ResNet-50 pour décomposer le problème en une combi-

naison de détection de région et de classification d'orientation (19 dans leur étude). Le premier, le Grasp Proposal Network, prend une image d'entrée et la traite à l'aide d'une succession de couches convolutionnelles et de max-pooling pour extraire des caractéristiques (features). Pour chaque position dans la carte des caractéristiques (feature map), le GPN prédit un ensemble de boîtes de délimitation (bounding box) qui sont centrées sur cette position. La seconde étape consiste à classer les propositions de régions prédites lors de l'étape précédente en fonction de l'orientation (une des orientations indiquant qu'une saisie n'est pas souhaitable), tout en affinant simultanément la boîte de délimitation de la proposition.

Des travaux récents tels que le Generative Grasping CNN (GG-CNN) [120] et le Generative Residual Convolutional Neural Network (GR-ConvNet) [89] utilisent des étiquettes conçues à la main pour générer une carte d'affordance de saisie par pixel. La tâche de saisie devient alors similaire à la segmentation sémantique, certaines parties de l'objet étant considérées comme saisissables tandis que d'autres doivent être évitées. Ces méthodes d'apprentissage de bout en bout présentent l'avantage de bénéficier des progrès continus dans les domaines de l'apprentissage profond et de la vision par ordinateur, permettant une meilleure intégration des informations brutes et une optimisation plus efficace de la performance de la préhension.

Une approche nécessitant un faible volume de données (Sec. 2.5.3.1), a été présentée par Hélénon et al. [64]. Ils ont proposé un pipeline CNN de segmentation pixel par pixel utilisant entre 1 et 3 démonstrations pour l'apprentissage. Ce pipeline est capable de prédire les emplacements de préhension autorisés à partir d'une image de profondeur tout en évitant les zones interdites. Les démonstrations sont obtenues en appliquant du ruban adhésif coloré sur les doigts de l'opérateur, ce qui permet de montrer des saisies parallèles. Les auteurs ont évalué leur pipeline sur 5 types d'objets industriels et ont atteint un taux de réussite compris entre 70 et 90% selon l'objet, bien que le pipeline soit limité à la saisie de 2 objets présentant 2 stratégies de préhension différentes avec le même réseau neuronal.

L'apprentissage par renforcement

L'apprentissage par renforcement (RL) consiste à apprendre une politique qui associe les états de l'agent aux actions à entreprendre pour maximiser les récompenses reçues en interagissant avec l'environnement à travers une méthode d'essai-erreur.

L'application de l'apprentissage par renforcement à la préhension d'objets pose de nombreux défis. L'espace d'exploration est si vaste que la réussite d'une saisie peut être peu probable, et il n'est pas certain que les actions du robot ne poseront pas de problèmes de sécurité. Cela conduit à un apprentissage lent, où la collecte de données peut être longue et où les échantillons positifs et négatifs sont fortement déséquilibrés.

Un exemple récent qui adresse ces défis est le travail de Zhu et al. [211], qui ont formulé le problème de la saisie en 4DOF comme un problème de bandit contextuel et ont utilisé l'apprentissage Q pour apprendre une politique en 600 essais sur 15 objets (validé sur 15 autres objets). Leur modèle utilise des réseaux neuronaux équivariants pour approximer la fonction de saisie SE(2)-équivariante, ce qui a permis d'améliorer l'efficacité de l'échantillonnage pour l'apprentissage de la préhension (Sec. 2.5.3.1).

Si l'apprentissage en simulation est utilisé pour surmonter certaines de ces difficultés, il est souvent nécessaire d'appliquer des méthodes d'adaptation pour être efficace dans le monde réel [15]. Des techniques comme l'apprentissage par transfert, la randomisa-

tion de domaine ou l'adaptation de domaine peuvent aider à réduire le fossé entre les environnements simulés et réels, améliorant ainsi la performance de l'apprentissage par renforcement dans des scénarios réels.

De plus, l'usage de l'apprentissage par renforcement profond, qui combine les réseaux de neurones profonds avec l'apprentissage par renforcement, a démontré des résultats prometteurs dans la résolution de problèmes complexes, y compris la préhension d'objets en robotique [175, 133]. En combinant l'apprentissage par renforcement avec d'autres méthodes d'apprentissage, comme l'apprentissage supervisé ou non supervisé, il est possible de tirer profit des avantages de chaque approche et de surmonter certaines des difficultés rencontrées lors de l'apprentissage de la préhension d'objets par des robots [207].

2.5.3 Vers des méthodes d'apprentissage de la préhension robotique plus adaptables

La préhension robotique est un domaine de recherche en pleine croissance qui a connu des progrès significatifs ces dernières années. Cependant, la plupart des méthodes actuelles nécessitent une quantité massive de données pour l'apprentissage, ce qui peut être coûteux en termes de temps et de ressources. De plus, ces méthodes ont souvent du mal à s'adapter à des tâches spécifiques ou à manipuler des objets de manière appropriée pour une tâche donnée. Dans cette section, nous explorons deux directions pour optimiser les méthodes d'apprentissage de préhension : la réduction de la dépendance aux données et l'orientation vers une tâche spécifique.

2.5.3.1 Techniques pour réduire la quantité de données nécessaires

Plusieurs travaux se sont concentrées sur l'apprentissage des préférences de préhension en exploitant un nombre restreint d'exemples [116], ou en adoptant des approches qui requièrent significativement moins de données que les méthodes conventionnelles [64, 210].

Le méta-apprentissage, basé sur l'imitation (Sec. 2.5.2.1), vise à acquérir des métaconnaissances à partir d'épisodes d'apprentissage antérieurs ou de différents domaines pour obtenir une méta-politique. Au cours de l'apprentissage par imitation, cette méta-politique est affinée grâce à la démonstration pour obtenir la politique finale. Cela permet de réaliser du few-shot et one-shot imitation learning [35, 9] à partir d'observations visuelles telles que des vidéos ou des images.

L'augmentation de données est une technique couramment utilisée pour traiter des images, comme dans [120, 89]. Elle consiste à générer de nouveaux exemples d'entraînement en appliquant des transformations telles que des rotations, des translations et des distorsions aux données existantes, rendant ainsi le modèle plus robuste face aux variations des entrées.

Une autre approche pour réduire la quantité de données nécessaires est l'inférence de préhensions SE(3) à partir de méthodes SE(2). Bien que la plupart des méthodes de préhension SE(3) soient basées sur des données de nuages de points 3D, certaines méthodes étendent des méthodes SE(2) pour obtenir des saisies en SE(3). Par exemple, [8] utilise un modèle entièrement convolutionnel pour déduire la position et l'orientation descendante d'une saisie, tandis que [38] utilise un algorithme d'amélioration itérative pour sélectionner la meilleure direction d'approche. De plus, [161] et [123, 80] génèrent des multiples vues virtuelles de l'objet pour faciliter la détection de la prise.

Ces méthodes SE(2) étendues sont intéressantes car elles permettent d'apprendre une capacité de préhension en 6-DOF à un rythme similaire à celui des méthodes de préhension planaire, tout en conservant une efficacité de calcul acceptable. Cependant, elles peuvent également présenter certaines limitations, comme une moindre robustesse face à des objets présentant des formes et des structures 3D complexes.

Enfin, le transfert d'apprentissage, qui consiste à entraîner un modèle sur un grand ensemble de données, puis à le peaufiner sur un ensemble de données plus petit et spécifique, est une autre méthode pour réduire la quantité de données nécessaires. En transférant les connaissances acquises sur le grand ensemble de données à l'ensemble de données plus petit, le transfert d'apprentissage peut réduire considérablement la quantité de données nécessaires pour obtenir de bonnes performances. Cette approche facilite l'adaptation rapide des modèles de préhension à de nouvelles tâches, contribuant ainsi à une meilleure efficacité en termes de données et à des performances améliorées dans diverses situations de manipulation d'objets.

En conclusion, les techniques pour réduire la quantité de données nécessaires pour l'apprentissage de la préhension robotique sont variées et comprennent le méta-apprentissage, l'augmentation des données, l'inférence de préhensions SE(3) à partir de méthodes SE(2), et le transfert d'apprentissage. Elles offrent des compromis entre la complexité et la performance, et peuvent permettre une plus grande flexibilité et une meilleure robustesse dans l'apprentissage de la préhension. Cependant, il est crucial de choisir la méthode appropriée en fonction des caractéristiques spécifiques de la tâche de préhension et des données disponibles.'

2.5.3.2 Préhension orientée vers une tâche spécifique

Certaines méthodes visent à adapter l'emplacement de la prise pour accomplir un objectif spécifique. Certains travaux antérieurs ont proposé de détecter automatiquement ces affordances [57] à partir de caractéristiques géométriques en utilisant des nuages de points 3D ou des images RGB-D [127, 163]. [85] a proposé un cadre pour classifier l'action de manipulation et l'objet impliqué dans la démonstration. De plus grandes bases de données d'étiquettes sémantiques [126, 3] catégorisent chaque partie d'un objet donné avec une fonctionnalité ou une affordance. Alors que la saisie générique d'objets inconnus nécessite l'utilisation de bases de données génériques, l'étiquetage de bases de données spécifiques pour des objets particuliers prend beaucoup de temps.

[124] ont fait appel au crowdsourcing pour la création de leur base de données TaskGrasp et utilisent la connaissance sémantique des objets et des tâches encodée dans un graphe de connaissances pour généraliser à de nouvelles instances d'objets, classes et tâches. Leur méthode se compose de 3 blocs : un encodeur de préhension et d'objets similaire à [122], un Graph Convolutional Network [84] qui prend en entrée la forme et la préhension encodées de l'objet ainsi qu'un graphe de connaissance encodant les relations sémantiques entre les catégories d'objets et les tâches, et un modèle évaluant les saisies.

La préhension orientée vers une tâche est une approche importante pour permettre aux robots d'interagir de manière plus efficace et adaptée avec leur environnement. En tenant compte des objectifs spécifiques et des affordances des objets, ces méthodes peuvent améliorer la performance de la préhension et faciliter l'exécution de tâches complexes. Cela permet aux robots de mieux s'adapter à des situations variées et de répondre aux besoins

spécifiques des utilisateurs ou des environnements dans lesquels ils opèrent.

2.6 Conclusion

Au cours des dernières années, l'importance croissante de l'apprentissage automatique a conduit à l'émergence de nouvelles techniques en matière de préhension robotique. Dans ces approches, les métriques de qualité de saisie sont acquises par des méthodes axées sur les données, plutôt que d'être prédéfinies. De nouveaux capteurs tactiles et visuels ont également été développés, accompagnant ces avancées et ouvrant la voie à des problèmes plus complexes, tels que :

— la saisie d'objets dont la forme n'est pas connue ;
— la saisie dans des tas d'objets variés et complexes ;
— l'utilisation d'actions non préhensiles avant la manipulation ;
— la détection de saisie en temps réel ou en boucle fermée ;
— la saisie orientée vers une tâche spécifique ;

Les méthodes classiques de classement des candidats à la préhension peuvent être lentes, car elles doivent évaluer un grand nombre de candidats. Toutefois, elles présentent l'avantage d'être plus facilement interprétables et intégrables dans des comportements plus complexes. Les méthodes basées sur l'imitation permettent d'apprendre des saisies spécialisées, jusqu'aux mouvements mêmes du robot, mais sont souvent les plus coûteuses en termes d'efforts demandés aux utilisateurs. L'inférence de préhension en une seule passe (avec des modèles convolutionnels entièrement convolutifs ou basés sur PointNet) raisonne efficacement sur les préhensions d'un objet ou d'une scène entière, permettant des performances en temps réel. Enfin, l'apprentissage de politiques offre la capacité d'apprendre des politiques en boucle fermée plutôt que de simplement détecter des poses de préhension, constituant un avantage majeur.

Les méthodes SE(2) sont comparées hors ligne sur la base de données de Cornell et avec les taux de succès de la préhension sur un robot réel. Elles sont en général basées sur des données d'images 2D, et les méthodes basées sur des modèles convolutionnels entièrement convolutifs semblent être les plus compétitives.

Les méthodes de préhension SE(3) sont basées sur des données de nuages de points 3D, utilisant des convolutions 3D ou des architectures PointNet pour traiter directement les nuages de points, ce qui est coûteux en calculs.

Les méthodes de préhension SE(3) ne disposent pas d'une base de données hors ligne standard, ce qui rend les obstacles à une comparaison équitable entre les différentes approches encore plus prononcés. ACRONYM [54] offre une possibilité récente, puisqu'il s'agit d'une base de données de préhensions en SE(3) comprenant 17,7 millions de saisies étiquetées effectuées sur 8 872 objets maillés. Ces objets maillés proviennent de ShapeNet [32], et les saisie sont été étiquetées à l'aide du simulateur de physique FleX pour simuler le préhenseur Franka Panda (avec une ouverture maximale de 8 cm).

Les taux de succès des préhensions proviennent d'expériences robotiques physiques qui dépendent des objets utilisés pour l'évaluation, de l'installation robotique physique, de la planification de mouvement, voire même du système de perception (par exemple, certaines méthodes fusionnent les données de plusieurs caméras en utilisant la méthode SLAM ou d'autres techniques). Un travail évaluant la réussite de la préhension en utilisant des objets

difficiles est désavantagé par rapport à un article qui utilise des objets plus simples.

La création de tests standardisés pour les méthodes de préhension a donc reçu de l'attention ces dernières années. Par exemple, GRASPA [13] propose une procédure expérimentale physique rigoureuse permettant d'évaluer les performances des pipelines de préhension. L'utilisation généralisée d'un ensemble fixe d'objets d'évaluation pour les expériences de préhension robotique physique faciliterait également les comparaisons. Des travaux tels que EGAD ! [118] ou AGOD-Grasp [2] pourraient fournir des objets d'évaluation imprimables en 3D avec une diversité suffisante.

2.6.1 Perspectives de contribution

Dans la suite de cette thèse, nous contribuons à la réflexion sur la préhension robotique en étudiant l'efficacité des approches existantes d'algorithmes de préhension (notamment sur les données de Cornell, YCB et AGOD) et en évaluant les performances humaines en matière de saisie d'objets dans un tas. Nous cherchons à mieux comprendre les stratégies et les mécanismes sous-jacents à la préhension humaine pour améliorer les algorithmes de préhension robotique et fixer des objectifs de performance auxquels les robots devraient aspirer.

Nous intégrons également les algorithmes que nous avons développés et utilisés à GRASPA, contribuant ainsi à l'amélioration de l'écosystème des méthodes de préhension robotique, en fournissant des solutions utilisables par d'autres chercheurs et praticiens et en facilitant la comparabilité entre les différentes approches.

Nous proposons également des techniques d'apprentissage data-efficace et orientées vers des tâches spécifiques, permettant aux robots d'acquérir rapidement des compétences de préhension et de manipulation dans divers scénarios et environnements, et de s'adapter aux nouvelles tâches avec un minimum d'exemples. Nous explorons l'intégration data efficient de l'expertise humaine pour guider l'apprentissage du robot et proposons une approche interactive de cet apprentissage.

De plus, nous étudions l'interaction entre les différentes modalités sensorielles (visuelles, tactiles, etc.) et leur intégration dans un cadre unifié d'apprentissage automatique pour la préhension robotique. Nous explorons l'utilisation de capteurs tels que le DIGIT pour améliorer la performance de la préhension robotique en ajoutant des informations tactiles basées sur la vision. Nous contribuons également à la caractérisation du capteur DIGIT et proposons un exemple d'utilisation via un régresseur capable de prédire la meilleure hauteur de saisie.

En résumé, nous contribuons au développement d'approches data-efficaces et orientées vers des tâches spécifiques, à l'étude de l'interaction entre les différentes modalités sensorielles et à leur intégration dans un cadre unifié d'apprentissage automatique. Nous soutenons également la mise en place de tests standardisés et la création d'ensembles d'objets d'évaluation fixes pour faciliter les comparaisons et améliorer les performances des robots dans des scénarios et environnements variés.

Chapitre 3

Apprentissage data-efficace des préférences de préhension.

Ce chapitre reprend, traduit et étend la publication :
(IEEE ICRA 2022) - Data-efficient learning of object-centric grasp preferences
Yoann Fleytoux, Anji Ma, Serena Ivaldi, Jean-Baptiste Mouret

Yoann Fleytoux a conçu et réalisé les expériences, écrit le logiciel, et préparé et révisé l'article.

Anji Ma (doctorant chinois en visite dans notre équipe) a contribué à la mise en place des expériences.

Serena Ivaldi a formulé le problème, supervisé et contribué à la conception expérimentale du travail de recherche, à la discussion et à l'interprétation des résultats, ainsi qu'à la rédaction et à la révision de l'article.

Jean-Baptiste Mouret a formulé le problème, supervisé et contribué à la conception expérimentale du travail de recherche, à la discussion et à l'interprétation des résultats, ainsi qu'à la rédaction et à la révision de l'article.

Les robots sont souvent chargés de saisir des objets d'une boîte ou d'un tapis roulant, notamment dans les milieux industriels [139]. Grâce aux récentes avancées de l'apprentissage profond, ils peuvent désormais saisir des objets inconnus avec des taux de réussite supérieurs à 90%. Pour ce faire, ils apprennent la relation entre la forme des objets et la saisie la plus appropriée.

Néanmoins, la meilleure saisie en fonction de la forme n'est pas toujours celle qui doit être privilégiée. Par exemple, un marteau doit généralement être pris par la tête car la distribution de la masse n'est pas uniforme, alors que la forme suggère qu'il devrait être pris par le milieu du manche. De même, un couteau tranchant doit généralement être saisi par le manche pour éviter d'endommager les doigts du robot, mais il peut être nécessaire de le saisir par la lame si le robot doit le donner à un opérateur. Pour divers

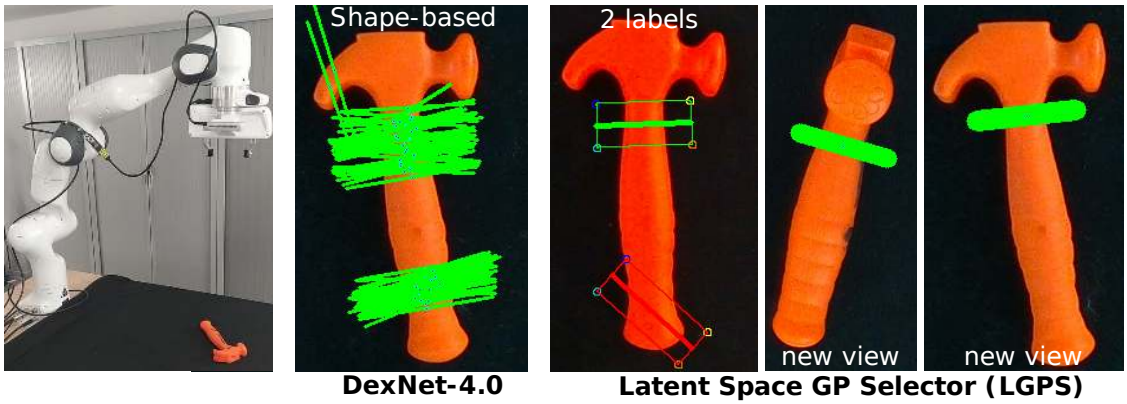


FIGURE 3.1 – Préhension générée par Dex-Net 4.0 pour un marteau jouet et préhension retenue avec notre méthode : seulement 2 étiquettes (une étiquette positive, en vert, et une étiquette négative, en rouge). Voir section 3.4.1.

objets, certaines méthodes de manipulation peuvent être déconseillées voire interdites, comme c'est le cas dans le secteur de l'industrie alimentaire. Dans d'autres situations, une manipulation incorrecte pourrait endommager ou altérer les objets, à l'instar de lunettes de vue qui risquent de se briser ou de se rayer si elles sont saisies par le verre. Il existe aussi des cas où la manipulation nécessite une attention particulière aux attributs spécifiques de l'objet, comme lorsqu'il s'agit de gérer une pièce enduite de peinture fraîche. Il est important de noter que la saisie qui devrait être privilégiée ne facilite pas toujours la réussite de celle-ci ; au contraire, il s'agit souvent d'une contrainte de la tâche.

Par conséquent, des connaissances expertes pour la saisie d'objets spécifiques sont nécessaires dans de nombreuses situations ; mais l'étiquetage de milliers d'images prend du temps et doit être effectué pour chaque application (par exemple, pour une usine de marteaux, puis pour un entrepôt de couteaux, etc.) Notre pipeline de préhension vise à apprendre des préférences de saisies centrées sur l'objet avec très peu d'exemples (typiquement 1 à 4 par objet) et à les généraliser à d'autres vues du même objet. Dans ce travail, nous ciblons des objets uniques sur un fond simple, par exemple des objets sur un tapis roulant dans un environnement industriel.

Notre idée principale est que nous pouvons utiliser des générateurs de saisies génériques pour rendre l'apprentissage des préférences efficace en termes de données de deux manières : (1) ils peuvent être exécutés sur une grande base de données d'images pour apprendre un espace latent de saisie, et (2) ils peuvent être utilisés pour générer des candidats de saisie de sorte qu'un classificateur basé sur les préférences n'ait qu'à choisir parmi les bonnes préhensions. En d'autres termes, un générateur de préhension extrait les aiguilles de la botte de foin, et un classificateur n'a plus qu'à choisir la meilleure aiguille dans un espace de faible dimension.

Notre hypothèse principale est que nous avons accès à une base de données d'images RGB-D obtenues passivement sur lesquelles un générateur de préhension basé sur la forme peut être exécuté ; nous considérons ce jeu de données comme "large et bon marché". Nous représentons les prises générées à l'aide de patchs d'images tournés et ajustés qui intègrent la prise et son contexte dans une seule entrée [106], ce qui permet d'apprendre une représentation à faible dimension des prises et de leur contexte à l'aide d'un autoencodeur

variationnel (VAE) [82, 20]. Nous utilisons ensuite cette représentation à faible dimension pour entraîner/interroger un classificateur à processus gaussien (GP) [66, 146] qui *filtre* les saisies générées par un générateur de saisies, qui sont souvent déjà efficaces, selon les préférences de l'expert. Il est important de noter que l'expert peut donner à la fois des "étiquettes positives", c'est-à-dire des saisies qui devraient être favorisées (par exemple, "c'est comme ça qu'il faut faire"), et des "étiquettes négatives", c'est-à-dire des saisies qui devraient être évitées (par exemple, "ne pas faire ça"). Nous appelons notre méthode "Latent Space GP Selector" (LGPS).

Une fois entraîné, notre pipeline génère des saisies candidates, les encode dans l'espace latent, puis interroge un classificateur à processus gaussien pour connaître la préférence de l'expert ; le robot exécute cette saisie avec un algorithme de planification standard [145]. Pour l'apprentissage des préférences, la saisie sélectionnée par l'expert est encodée dans le même espace latent et le classificateur à processus gaussien est mis à jour.

Alors que le VAE et le GP ont été combinés ensemble dans différents domaines (par exemple, [200] pour les vidéos), notre principale contribution est la combinaison de générateurs de préhension (qui peuvent, par exemple, être basés sur l'apprentissage profond) avec une représentation de préhension basée sur l'image pour apprendre un espace latent de préhension de manière non supervisée. Nous montrons que notre pipeline permet d'apprendre des saisies qui sont environ cohérentes à 80% avec les étiquettes des experts avec moins d'un exemple par objet sur le jeu de données Cornell [75] (885 scènes, 244 objets) et sur notre propre jeu de données (91 scènes, 28 objets).

3.1 Définition du problème

Nos principales hypothèses sont les suivantes : (1) le robot dispose d'une caméra RGB-D ; (2) tous les objets peuvent être saisis par le haut ; (3) nous avons accès à une grande base de données non étiquetées d'images RGB-D (**D**) ; (4) nous avons accès à une petite base de données étiquetées (≤ 4 par objet) de saisies positives (bonne saisie) ou négatives (saisie à éviter) (base de données **E**).

Notre objectif principal est d'apprendre à reproduire les saisies que l'expert a étiquetées comme bonnes pour de nouvelles vues d'objets qui ont été vus précédemment. Notre objectif secondaire est de généraliser à des objets qui n'ont jamais été vus mais qui sont proches de ceux déjà étiquetés.

Nous évaluons la performance en utilisant la métrique *rectangle* [75], ce qui permet de comparer avec des travaux précédents utilisant des jeux de données publiés [75, 49]. Pour cette métrique, les saisies sont représentées par des rectangles centrés sur la position centrale de la pince (x, y), tournés selon l'orientation $\theta \in [-\frac{\pi}{2}, +\frac{\pi}{2}]$, avec une largeur égale à l'ouverture de la pince l, et une hauteur qui code la tolérance. Deux saisies sont comparées en regardant à quel point les deux rectangles correspondants se chevauchent. Plus précisément, étant donné une préhension proposée GC et une préhension de base GT, l'intersection sur l'union (IoU, également appelée indice de Jaccard) correspond à la zone d'intersection normalisée :

$$IoU(GT, GC) = \frac{|GT \cap GC|}{|GT \cup GC|} \tag{3.1}$$

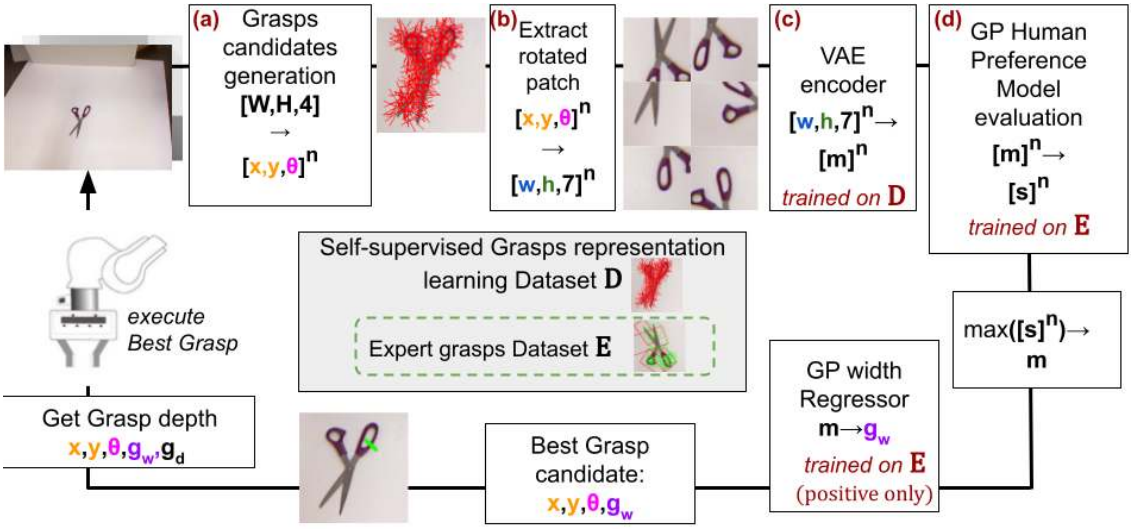


FIGURE 3.2 – Le pipeline de saisie LGPS, en supposant que le VAE ait été entraîné précédemment (Sec. 3.3.3). À partir d'une image RGB-D, un générateur de saisie (Sec. 3.3.1) crée des candidats à la saisie sous forme de segments. Ces candidats à la saisie sont représentés par des patchs tournés centrés sur le milieu du segment (Sec. 3.3.2). Chacun d'entre eux est soumis à un VAE pour obtenir sa représentation latente, qui est, à son tour, l'entrée du classificateur GP (Sec. 3.3.4) afin d'obtenir la probabilité estimée d'être sélectionné par l'expert ; la saisie avec la probabilité la plus élevée est sélectionnée. Un deuxième GP est interrogé pour obtenir la largeur de la pince pour la prise sélectionnée. La "profondeur" de la prise (la position z de la pince) est calculée en utilisant l'image de profondeur.

La métrique du rectangle RM est égale à 1 si cette zone de chevauchement est supérieure à 0.25 et si la différence angulaire est inférieure à $\frac{\pi}{6}$:

$$RM(GT, GC) = \begin{cases} 1, & \text{if } IoU(GT, GC) > 0.25 \\ & \text{and } |GT.\theta - GC.\theta| \leq \frac{\pi}{6} \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

Nous comparons nos résultats à ceux de GR-Convnet [89] et de GC-CNN [120] avec la métrique des rectangles sur le Cornell Dataset [75] et sur notre propre jeu de données, qui se concentre sur les étiquettes significatives spécifiques aux objets (ciseaux, marteau, etc.). Notre méthode n'apprend pas de hauteur pour les rectangles, nous l'avons donc fixée à 38 pixels.

Bien que le processus décrit dans ce chapitre se déroule en mode hors ligne, avec l'apprentissage réalisé à partir d'une base de données étiquetées ou d'un sous-ensemble de celle-ci, nous avons mis en œuvre une configuration *en ligne* (voir chapitre 6). Dans ces scénarios, l'expert n'intervient pour corriger le robot qu'en cas d'erreur.

3.2 Prérequis

3.2.1 Autoencodeur (AE)

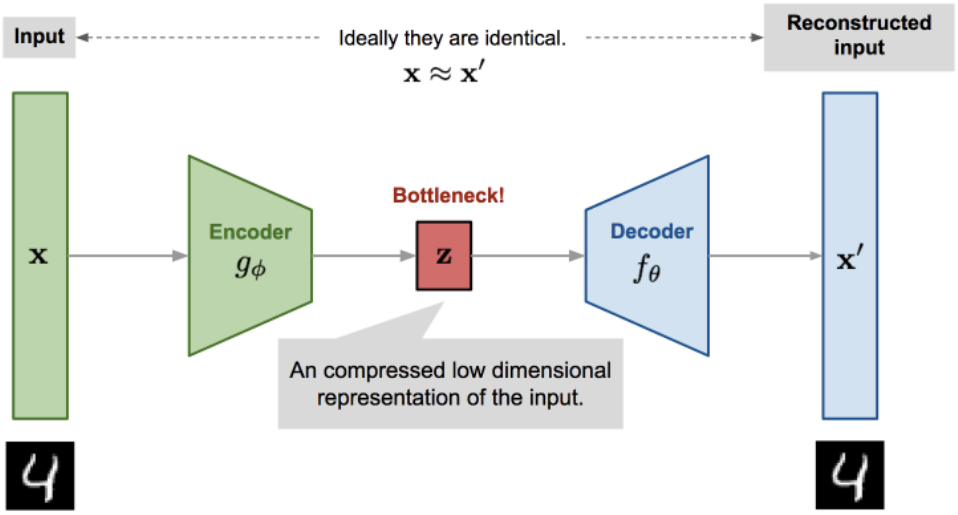


FIGURE 3.3 – Schéma de l'autoencodeur, figure de [192].

Les autoencodeurs (AE) sont des réseaux de neurones composés de deux réseaux séquentiels : l'encodeur et le décodeur. L'encodeur compresse les données de haute dimension x en données de plus basse dimension z. L'espace des vecteurs compressés z est appelé espace latent. L'encodeur est généralement qualifié de "goulot d'étranglement" car il doit apprendre une compression efficace des données dans cet espace de dimension inférieure. Le décodeur est un autre réseau neuronal, il prend en entrée l'espace latent z et renvoie

une représentation x' de même dimension que x ; x' représente la reconstruction de x (Fig. 3.3).

Cet apprentissage n'a pas besoin de données étiquetées ; il transforme un tenseur de haute dimension en une représentation plus dense, avec idéalement la même quantité d'informations représentée par moins de bits, ce qui est utile pour divers usages, tels que la compression des données ou la mise en évidence des facteurs génératifs sous-jacents des données.

La fonction de perte est généralement l'erreur quadratique moyenne ou l'entropie croisée entre la sortie et l'entrée, ce qui pénalise la différence entre la sortie reconstruite et l'entrée.

Un problème des autoencodeurs classiques est que l'espace latent dans lequel les entrées sont encodées peut ne pas être continu et ne permet donc pas une interpolation facile (générer des données à partir de variables latentes aléatoires générera des résultats inutiles). C'est pourquoi des autoencodeurs variationnels (VAE) ont été proposés.

3.2.2 Autoencodeur Variationnel (VAEs)

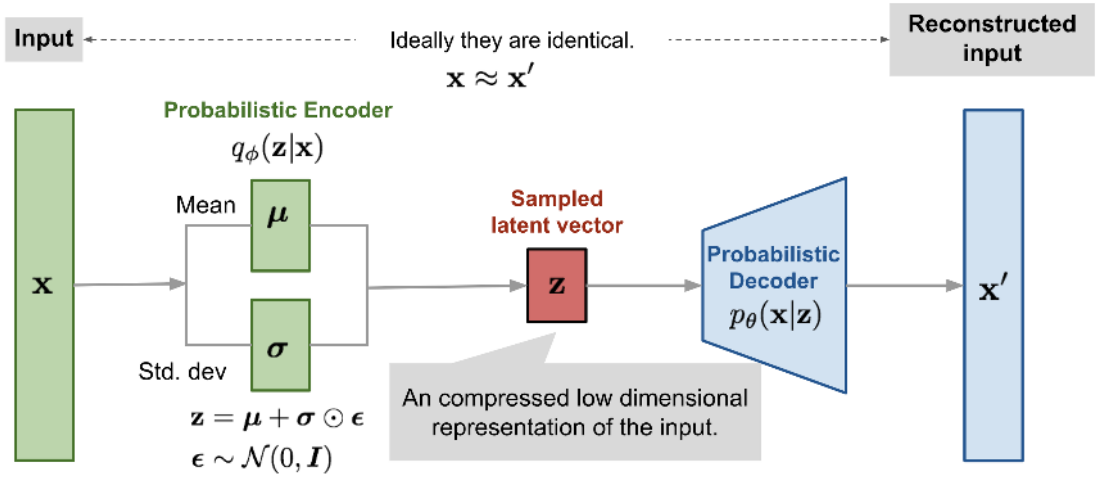


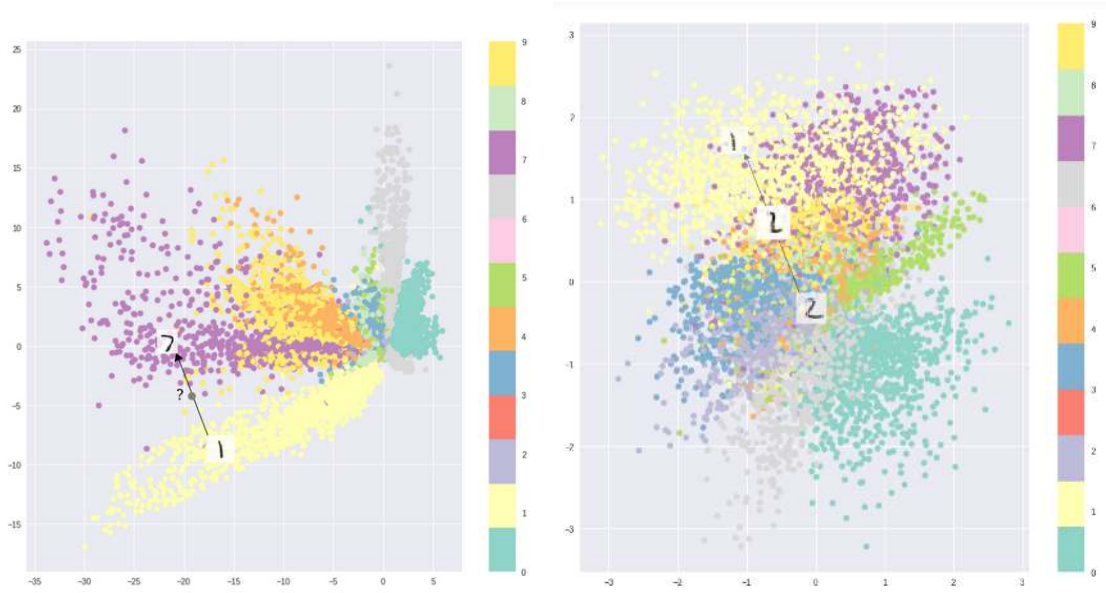
FIGURE 3.4 – Schéma de VAE, figure de [192]

Les VAE ont une propriété fondamentalement unique qui les sépare des simples autoencodeurs : leurs espaces latents sont par conception continus. Au lieu de convertir les données en entrées en un vecteur fixe, un VAE les convertit en une distribution (Fig. 3.4).

Ils le font en créant deux vecteurs μ, σ qui représentent la moyenne et l'écart type. Le vecteur latent de l'entrée x est échantillonné dans la région $\mathcal{N}(\mu, \sigma)$. Ainsi, lors de l'entraînement, plusieurs z peuvent représenter le même x.

Le vecteur moyen contrôle l'endroit où l'encodage d'une entrée doit être centré, tandis que l'écart type contrôle la "zone", c'est-à-dire l'écart que l'encodage peut avoir par rapport à la moyenne. Comme les encodages sont générés de manière aléatoire à partir de n'importe quel endroit à l'intérieur du "cercle" (la distribution), le décodeur apprend

que non seulement un point unique dans l'espace latent se réfère à un échantillon, mais que tous les points voisins s'y réfèrent également.



(a) Espace latent avec AE

(b) Espace latent avec VAE

FIGURE 3.5 – Dans ces graphiques, nous pouvons visualiser comment chaque image de MNIST (base de données d'images de chiffres manuscrits (0-9) en niveaux de gris 28x28, composée de 60 000 images d'entraînement et de 10 000 images de test), est transformée dans un espace latent de dimension deux, à la fois pour l'autoencodeur et VAE. On observe que l'espace latent de l'autoencodeur est beaucoup moins centré que celui du VAE, les 'groupes' pour l'autoencodeur sont également plus imbriqués et étalés. Comme aucune structure ou contrainte n'est imposée sur l'espace latent de l'autoencodeur, il est difficile d'interpréter ce que représente une certaine partie de son espace latent (voir le ' ?' entre le groupe des 1 et le groupe des 2 sur la figure (a)). Pour l'espace latent du VAE, les données entre deux groupes dans l'espace latent représentent une interpolation continue des informations encodées (ce qui est apparent pour la transition entre le groupe des 1 et le groupe des 2 dans la figure (b)).

Idéalement, ces représentations sont toutes aussi proches que possible les unes des autres tout en étant distinctes, permettant une interpolation fluide et la génération de nouveaux échantillons (Fig. 3.5).

Pour forcer cela, on introduit la divergence de Kullback-Leibler (divergence KL) dans la fonction de perte. La divergence de KL entre deux distributions de probabilité mesure simplement à quel point elles divergent l'une de l'autre.

$$\sum_{i}^{n}(\sigma_i^2 + \mu_i^2 - log(\sigma_i) - 1) \tag{3.3}$$

Pour rendre le modèle entraînable avec la descente de gradient, l'astuce de reparamétrisation est introduite, ce qui rend le processus d'échantillonnage du VAE différentiable

en introduisant une variable aléatoire auxiliaire ϵ à partir d'une gaussienne unitaire, puis en décalant l'échantillon aléatoire ϵ par la moyenne μ de la distribution latente et en le mettant à l'échelle par la variance σ de la distribution latente.

Grâce à cette reparamétrisation, nous pouvons maintenant optimiser les paramètres de la distribution tout en conservant la possibilité d'échantillonner aléatoirement à partir de cette distribution (Fig. 3.6).

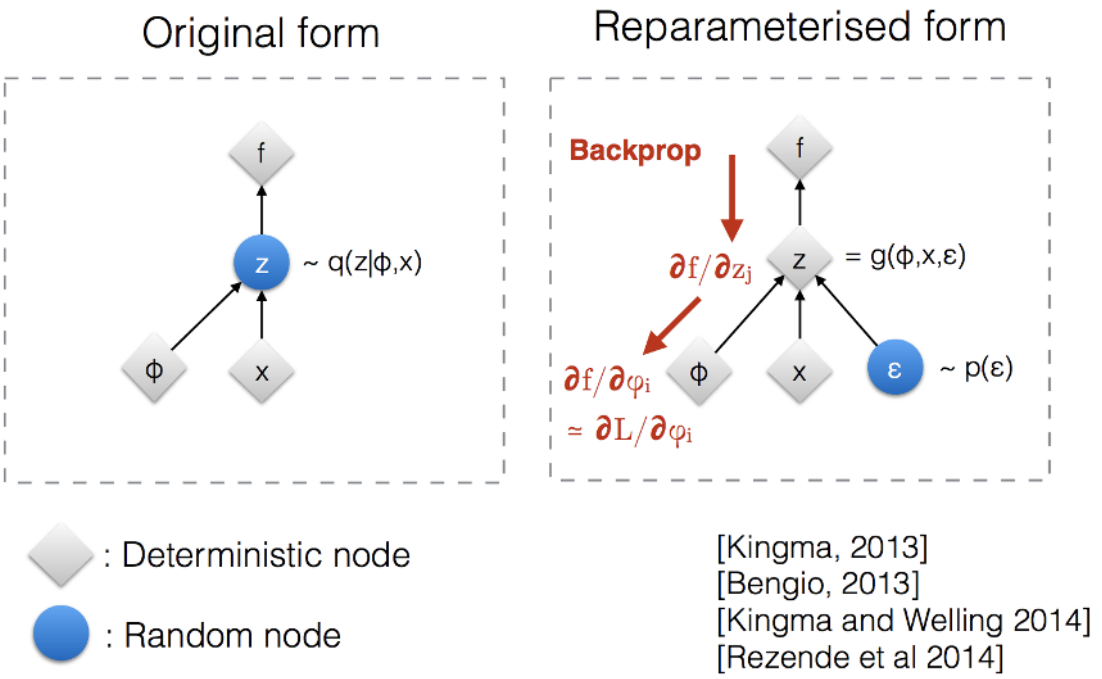


FIGURE 3.6 – Illustration de la manière dont l'astuce de reparamétrisation rend l'échantillonnage z entraînable, figure de [83]

Les VAE sont largement utilisés dans l'apprentissage semi-supervisé, les outils de réduction du bruit et les réseaux génératifs. En particulier pour les réseaux génératifs, ils utilisent les propriétés encodées dans l'espace latent : par exemple, si nous voulons appliquer une caractéristique à une image donnée, nous déplaçons sa représentation dans la direction dans laquelle ce type de caractéristique est encodé.

3.2.3 Processus Gaussiens

Les processus gaussiens (GP) [146] sont des modèles probabilistes qui peuvent être utilisés pour les tâches de régression et de classification. Ils constituent une généralisation de la distribution gaussienne, qui est utilisée pour modéliser la distribution d'une fonction à valeur continue $f(x)$ qui associe à chaque $\mathbf{x}$ une distribution gaussienne avec une moyenne μ et une variance σ.

$$\hat{f}_d(\mathbf{x}) \sim \mathcal{GP}(\mu(\mathbf{x}), k(\tilde{\mathbf{x}}, \mathbf{x}')) \tag{3.4}$$

Comme toutes les méthodes bayésiennes, les GP commencent par une distribution antérieure et la mettent à jour au fur et à mesure que des points de données sont observés, produisant ainsi la distribution postérieure des fonctions.

Les GP sont définis par une fonction moyenne $m(x)$ et une fonction de covariance (ou fonction noyau) $k(x, x')$. La fonction moyenne définit la valeur attendue de la fonction, tandis que la fonction de covariance décrit comment deux points s'influencent mutuellement en fonction de leur distance. La démarche courante est de diminuer exponentiellement l'influence avec la distance euclidienne (c'est la fonction noyau exponentiel-carré) :

$$k(x_1, x_2) = \sigma^2 \exp\left(-\frac{||x_1 - x_2||^2}{2\ell^2}\right) \tag{3.5}$$

Où σ^2 est la variance globale (ou l'amplitude) et ℓ^2 est l'échelle de longueur. Il s'agit de deux hyperparamètres qui peuvent être appris à partir de données.

Entraîner un modèle avec un processus gaussien revient à conditionner le processus gaussien par les données observées. En supposant que $D^d_{1:t} = \{f_d(\mathbf{x_1}), ..., f_d(\mathbf{x}_t)\}$ est un ensemble d'observations (résultats de l'évaluation de la fonction de récompense inconnue à modéliser $f(x)$), le GP peut être interrogé à un nouveau point d'entrée $\mathbf{x}_*$:

$$p(\hat{f}(\mathbf{x}*)|D^d 1 : t, \tilde{\mathbf{x}}*) = \mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x}_)) \tag{3.6}$$

Le calcul du vecteur noyau $\boldsymbol{k} = [kernel(\mathbf{x}_1, \mathbf{x}), \cdots, kernel(\mathbf{x}_t, \mathbf{x})]$ est nécessaire, où $kernel(\mathbf{x}, \mathbf{y})$ est la fonction noyau et une matrice noyau K, avec des entrées $K^{ij} = kernel(\mathbf{x}_i, \mathbf{x}_j)$

$$K = \begin{bmatrix} k(x1_1, x_1) & \cdots & k(x_1, x_t) \\ \vdots & \ddots & \vdots \\ k(x_t, x_1) & \cdots & k(x_t, x_t) \end{bmatrix} + \sigma^2_{noise} I \tag{3.7}$$

La moyenne $\mu(x)$ et la variance $\sigma(x)$ peuvent ainsi être calculées :

$$\mu(\mathbf{x}) = \boldsymbol{k}^T K^{-1} D^d_{1:t} \; \sigma(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - \boldsymbol{k}^T K^{-1} \boldsymbol{k} \tag{3.8}$$

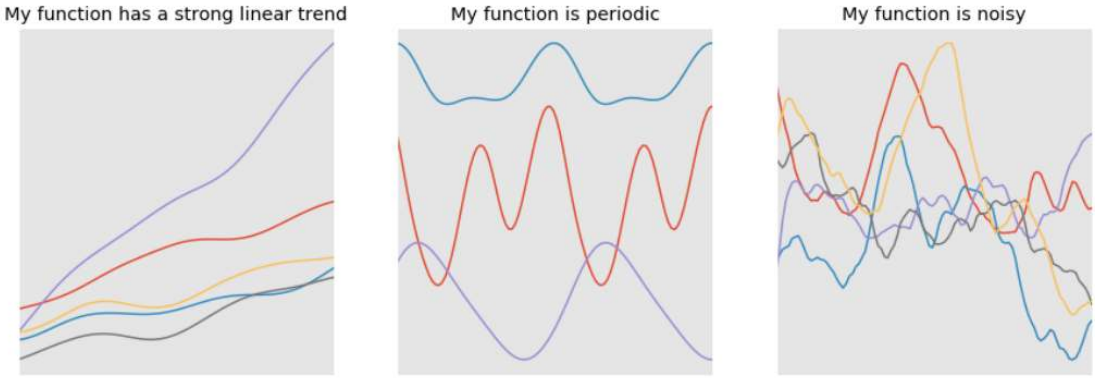


FIGURE 3.7 – Si l'on dispose d'une certaine connaissance des propriétés de la fonction cible, il est possible de l'intégrer au modèle.

Diverses fonctions de noyau existent pour modéliser un large éventail de fonctions (par exemple, un noyau périodique avec des fonctions sin/cos). Les noyaux courants comprennent l'exponentielle au carré, les noyaux périodiques et les noyaux quadratiques rationnels. Il est donc possible d'ajouter des connaissances préalables et des spécifications sur la forme du modèle en sélectionnant différentes fonctions de noyau (Fig. 3.7).

Un avantage des processus gaussiens, en particulier par rapport aux réseaux de neurones, est l'existence de cette covariance. Celle-ci peut être utilisée comme une mesure de l'incertitude du modèle concernant les prédictions faites.

Les GP peuvent également modéliser des fonctions non stationnaires, gérer des vraisemblances non gaussiennes, des fonctions à sorties multiples et l'optimisation bayésienne et fonctionnent également bien avec peu de données.

Par rapport aux réseaux neuronaux, les classificateurs à processus gaussiens sont plus précis lorsqu'il y a peu de données, au prix d'un temps de requête/d'entraînement plus long (la requête est $O(n^2)$ avec n le nombre d'échantillons).

Pour la classification GP, l'objectif est de modéliser la probabilité d'appartenance à une classe pour une entrée donnée. Les deux approches principales pour la classification par GP sont "un contre tous" et les modèles à variables latentes.

L'approche "un contre tous" consiste à ajuster un GP pour chaque classe en considérant cette classe comme positive et toutes les autres comme négatives. Pour chaque entrée, la probabilité postérieure est calculée pour chaque classe, puis la classe ayant la probabilité la plus élevée est choisie comme prédiction.

L'approche à variables latentes consiste à introduire une variable latente pour représenter les étiquettes de classe. Le modèle est alors construit pour modéliser la relation entre les variables d'entrée, la variable latente et les données observées. Les prédictions sont alors obtenues en inférant la valeur de la variable latente pour chaque entrée et en utilisant cette valeur pour calculer la probabilité postérieure de chaque classe.

Les modèles SVGP (Sparse Variational Gaussian Processes) [65] sont une variante des GP qui permettent de réduire le coût de calcul et de stockage associé à l'utilisation de GP pour les grandes ensembles de données.

L'approche SVGP consiste à utiliser une version approximative de la fonction de co-

variance du GP, qui est représentée par un petit ensemble de points d'ancrage (appelés "inducteurs" ou "points de support") au lieu de tous les points de données. Les prédictions sont alors obtenues en utilisant une forme de variational Bayes pour inférer une approximation de la distribution postérieure du GP à partir des données et des points d'ancrage.

Cela permet de réduire la complexité de calcul du modèle GP et de rendre l'apprentissage possible pour les ensembles de données volumineux. En outre, l'approche SVGP permet également de régulariser le modèle, ce qui peut aider à améliorer la généralisation et à éviter le surapprentissage.

En résumé, les processus gaussiens sont des modèles probabilistes flexibles et puissants qui peuvent être utilisés pour la régression et la classification. Ils offrent plusieurs avantages, tels que la possibilité d'intégrer des connaissances préalables et des spécifications sur la forme du modèle, et une mesure de l'incertitude concernant les prédictions. Les processus gaussiens sont particulièrement adaptés aux problèmes avec des données limitées, mais peuvent être coûteux en termes de calcul et de stockage pour les grands ensembles de données. Les variantes comme les modèles SVGP aident à atténuer ces problèmes en réduisant la complexité du modèle.

3.3 Méthode

Notre pipeline algorithmique complet est illustré dans (Fig. 3.2). Les sous-sections suivantes fournissent des détails sur les composants respectifs (a-d) du pipeline.

3.3.1 Génération des candidats de saisie (a)

Étant donné une image RGB-D, les candidats à la saisie peuvent être générés à l'aide de techniques de vision par ordinateur [181], ou bien en les échantillonnant de manière aléatoire, ou en utilisant des méthodes basées sur l'apprentissage profond comme Dex-Net [108], Generative Grasping CNN (GG-CNN) [120] ou Generative Residual Convolutional Neural Network (GR-ConvNet) [89]. Nous utilisons ici un générateur de préhension basé sur la vision par ordinateur, qui s'est révélé à la fois rapide et efficace dans nos expériences, une fois combiné à un classificateur à processus gaussien (Sec. sec :GP), bien qu'un générateur à apprentissage profond pourrait facilement prendre sa place.

Nous extrayons d'abord l'objet avec l'algorithme GrabCut [151] qui fonctionne comme une "baguette magique" pour séparer un objet de son arrière-plan en utilisant quatre classes de pixels : provenant des objets, probablement de l'objet, pas de l'objet, et probablement pas de l'objet. Nous déterminons la classe de chaque pixel à l'aide de trois méthodes : segmentation colorée basée sur le HSV, segmentation basée sur la saillance [70], segmentation arrière-plan/fond basée sur le mélange gaussien [78]. Nous exécutons un détecteur d'arêtes Canny [30] pour obtenir les arêtes et nous calculons le squelette [94].

Nous générons des candidats à la saisie en calculant des lignes perpendiculaires à chaque pixel de bord et à chaque pixel de squelette [181]. Pour ce faire, on examine les deux voisins les plus proches de chaque point des bords/squelette. Pour maintenir le nombre de candidats à la saisie à un niveau faible, nous sautons les points d'arête/squelette si la distance avec le point précédent est inférieure à 4 pixels (base de données Cornell)

et à 8 pixels (notre base de données). Enfin, nous ajoutons un angle aléatoire (± 0.4 rad, distribution gaussienne) à chaque candidat pour augmenter la diversité des saisies. À ce stade, les saisies n'ont pas de largeur de préhension puisqu'elles sont utilisées pour (1) générer des patches (Sec. 3.3.2) qui ont une taille fixe (donc seules la position et l'orientation comptent), et (2) pour interroger un GP, qui prédit la largeur de préhension en utilisant la représentation latente du patch.

Ces n points et angles $[x, y, \theta]^n$ constituent la liste des candidats à la saisie pour une image RGB-D spécifique.

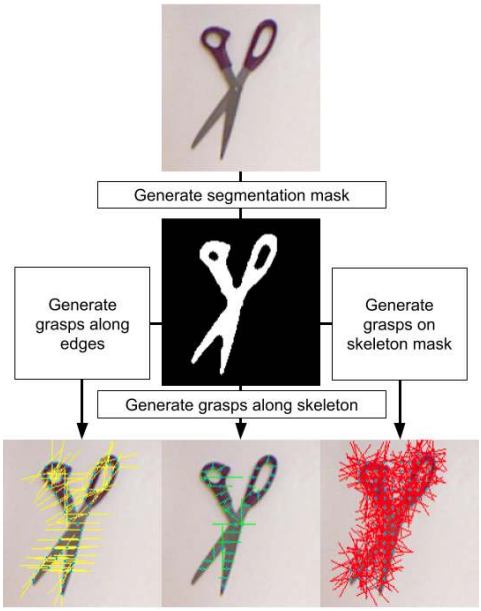


FIGURE 3.8 – Génération de candidats à la saisie à partir des images RGB. Nous extrayons un masque, puis les bords et le squelette.

3.3.2 Représentation des candidats de saisie (b)

Nous encodons les prises en tant que *image patch* à 7 canaux (Fig. 3.9), comme dans [138] et [106], qui centre chaque image sur sa prise respective. Le principal avantage de cette représentation est qu'elle combine une spécification de la prise (orientation, position par rapport à l'objet) avec une image de l'objet pour le reconnaître. Ces patchs d'images peuvent être facilement introduits dans des réseaux de neurones convolutifs, contrairement à une représentation basée sur des coordonnées ou des caractéristiques, qui devrait être associée au bon objet.

Pour créer ces patchs à partir de $[x, y, \theta]$ (Sec. 3.3.1), nous.. : (1) faisons pivoter l'image pour qu'elle corresponde à l'orientation du segment, (2) traduisons l'image pour qu'elle soit centrée sur le centre du segment, (3) recadrons l'image à 128×128 pixels. La largeur de la pince est ignorée pour l'étape d'extraction du patch. Nos patchs d'image sont à 7 canaux : 3 canaux pour l'image RGB, 1 canal pour la profondeur et 3 canaux pour l'image de la normale à la surface, générée à partir de l'image en échelle de gris de la profondeur.

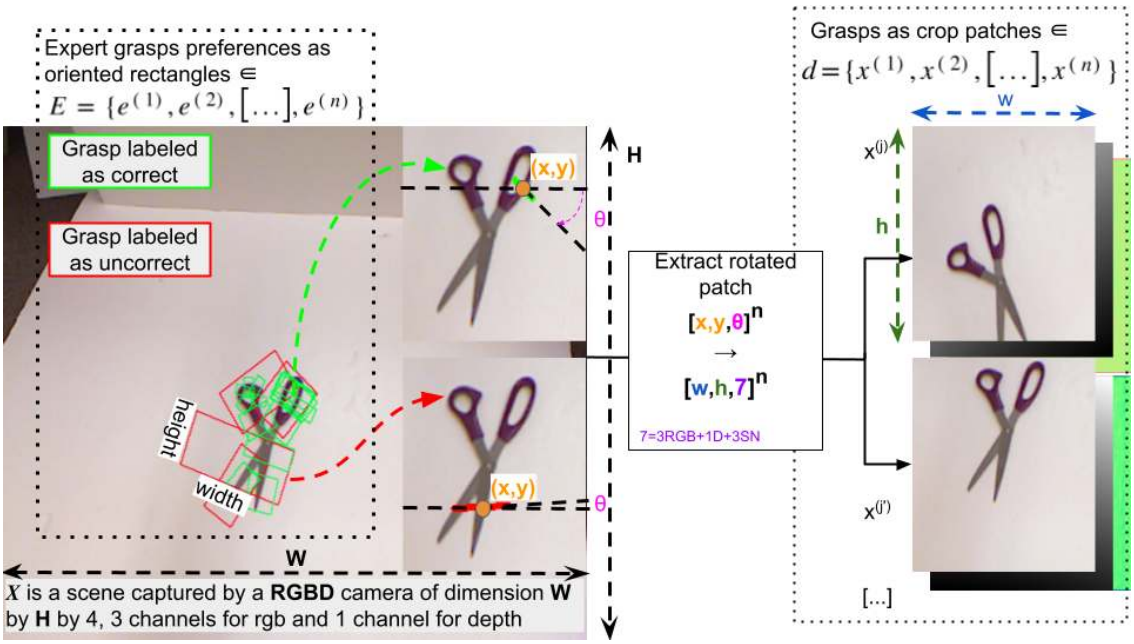


FIGURE 3.9 – Les saisies sont représentées comme un patch de l'image tourné et recadré qui est centré sur la saisie. Cette représentation permet d'alimenter les réseaux de neurones convolutifs avec les prises et leur contexte. .

3.3.3 Espace latent de préhensions (c)

Nous supposons ici que nous avons accès à un grand jeu de données de patchs (appelé d) qui n'est pas étiqueté. Nous entraînons un décodeur automatique convolutif à variables β [67, 20] en utilisant un grand nombre de patchs (au moins 40000, selon le jeu de données) générés à partir d'images RGB-D. La dernière couche du décodeur utilise une fonction d'activation tanh parce que nous normalisons la fonction d'activation. La dernière couche du décodeur utilise une fonction d'activation tanh car nous normalisons notre entrée à [-1, 1]. Compte tenu du VAE entraîné, l'extraction des patchs de saisie et leur encodage dans l'espace latent prennent moins de 2 secondes par scène en moyenne sur notre ordinateur.

3.3.4 Apprentissage du modèle de préférence des expert (d) avec des Processus Gaussien

Nous supposons maintenant que nous avons accès à un second jeu de données, pour lequel un petit ensemble de patchs a été étiqueté comme positif (sélectionné par l'expert) ou négatif (saisie à éviter). Pour les exemples positifs, le jeu de données contient également la largeur d'ouverture de la pince sélectionnée par l'expert.

Nous entraînons un classificateur gaussien [146] qui prend en entrée le code latent d'un patch, c'est-à-dire un candidat à la saisie, et produit un score s entre 0 et 1 qui décrit la probabilité que la saisie soit choisie par l'expert. En utilisant l'exemple positif, nous entraînons également le régresseur de processus gaussien [146] qui prend la même entrée et sort une distribution de probabilité de la largeur de la pince. Dans nos tests, nous avons observé un temps d'interrogation de moins de 0,06 seconde par scène.

Pour chaque saisie générée (Sec. 3.3.1), nous générons d'abord le patch correspondant, nous l'encodons dans l'espace latent en utilisant le décodeur β-VAE, puis nous interrogeons le GP pour obtenir son score. Nous sélectionnons le candidat à la saisie ayant le score le plus élevé. Pour adapter le classificateur GP à de nombreux échantillons sans compromettre le temps d'interrogation, nous utilisons des "classificateurs à processus gaussiens variationnels évolutifs" [66], qui sont basés sur un cadre de points inducteurs variationnels, bien que n'importe quel classificateur GP puisse être utilisé à la place (pour moins de 500 étiquettes environ, un classificateur GP standard fonctionne bien). La largeur d'ouverture de la pince g_w est sélectionnée en interrogeant le régresseur GP avec le patch sélectionné (un GP standard est utilisé pour cela car nous n'avons besoin que d'une seule requête, par rapport aux nombreuses requêtes pour le classificateur).

Pour utiliser la représentation rectangulaire orientée (voir Fig. 3.9), nous utilisons une valeur *height* fixe. Lors de l'exécution de la saisie sélectionnée avec le robot, la profondeur est calculée en utilisant les données de profondeur de la caméra RGB-D : nous extrayons un patch orienté du nuage de points de profondeur avec la largeur de la saisie sélectionnée et une hauteur fixe (5 pixels), et nous utilisons le point le plus proche de la pince g_d (c'est-à-dire le point le plus haut de l'objet) comme référence z.

3.4 Évaluation expérimentale

Nous évaluons notre approche sur le jeu de données Cornell [96] (885 scènes, 244 objets, 5055 étiquettes positives, 2822 étiquettes négatives) et sur un jeu de données personnalisé pour lequel les humains ont généralement des préférences (Fig. 3.12, 91 scènes, 28 objets, 447 étiquettes positives, 145 étiquettes négatives). Pour le jeu de données Cornell, le VAE (Tableau supplémentaire 3.1) est entraîné sur les 885 scènes (244 objets). Pour notre jeu de données, les images sont obtenues avec une caméra de profondeur Intel RealSense D415 montée sur la pince d'un robot Franka-Emika Panda (Fig. 3.1) qui est positionnée à 65 centimètres au-dessus des objets. Les objets sont pour la plupart issus du jeu de données YCB [28]. Nous avons utilisé un espace latent à 32 dimensions (sortie du VAE et entrée du GP).

Nous nous concentrons sur la performance avec très peu d'étiquettes (moins de 4000, idéalement moins de 50) car nous envisageons des scénarios interactifs ou semi-interactifs dans lesquels un expert ne veut pas passer beaucoup de temps à étiqueter. De plus, nous sommes intéressés par la généralisation à de nouvelles vues d'objets pour lesquels nous avons des étiquettes, et pas nécessairement à de nouveaux objets qui n'ont jamais été vus. Par conséquent, nous ne faisons aucun effort pour diviser la base de données d'entraînement et la base de données d'apprentissage en ensembles d'objets disjoints ; au contraire, nous attendons généralement de notre algorithme qu'il sélectionne la bonne saisie avec 1 ou 2 exemples de saisie du même objet dans des vues différentes. Par exemple, nous n'attendons pas de notre algorithme qu'il sache comment saisir un marteau s'il n'en a jamais vu, mais nous voulons qu'il apprenne à saisir un marteau de n'importe quel point de vue une fois qu'on lui a expliqué comment le faire une ou deux fois.

Nous comparons notre méthode à deux méthodes de préhension de l'état de l'art basées sur des étiquettes d'experts : Generative Grasping CNN (GG-CNN) [120] et Generative Residual Convolutional Neural Network (GR-ConvNet) [89] en utilisant la métrique du

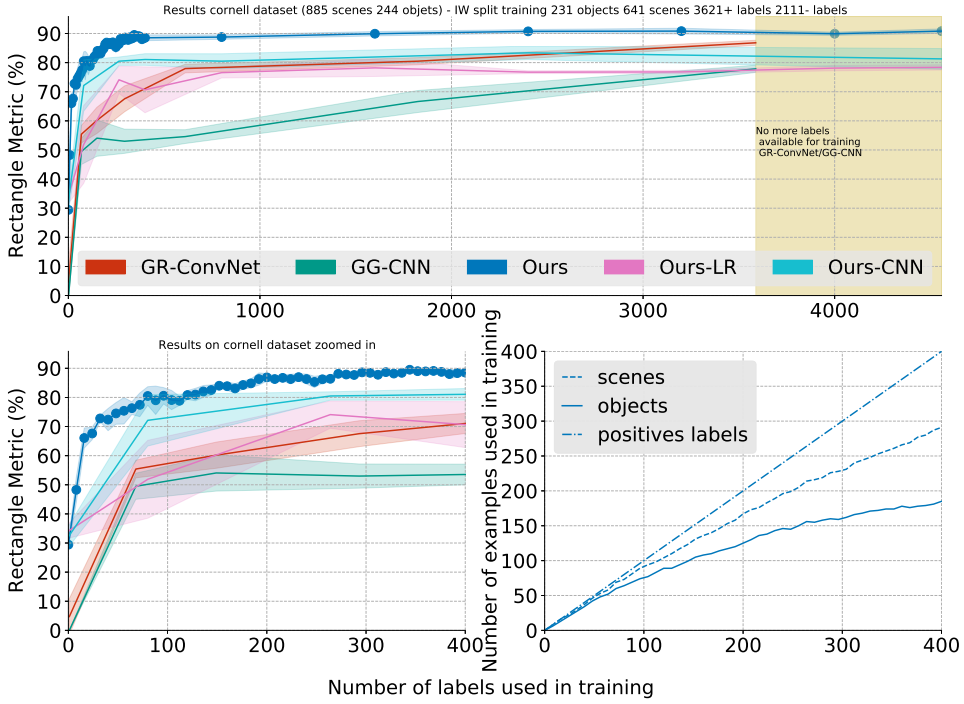


FIGURE 3.10 – Métrique du rectangle pour le jeu de données Cornell. L'axe x correspond au nombre d'étiquettes utilisées pour l'apprentissage (données d'apprentissage+validation, 641 scènes) et l'axe y à la métrique du rectangle en pourcentage du nombre de scènes dans la base de données de test (244 scènes). La métrique du rectangle compte le nombre de fois où la saisie sélectionnée correspond à au moins une des étiquettes positives. Pour la base de données Cornell, le graphique du haut montre la métrique du rectangle de 0 à toutes les étiquettes disponibles (GR-ConvNet et GG-CNN n'utilisent pas les étiquettes négatives, ils ne peuvent donc pas utiliser toutes les étiquettes), le graphique en bas à gauche correspond aux mêmes données mais se concentre sur les résultats de 0 à 400 d'étiquettes, et le graphique en bas à droite rapporte le nombre de scènes et d'objets par rapport au nombre d'étiquettes utilisées pour l'entraînement. Pour mieux comprendre les performances de notre pipeline, nous comparons deux "ablations" : remplacer le classificateur GP par un modèle de régression logistique pour la classification binaire ("Ours-LR") et utiliser un classificateur binaire CNN au lieu de la combinaison VAE+GP ("Ours-CNN").

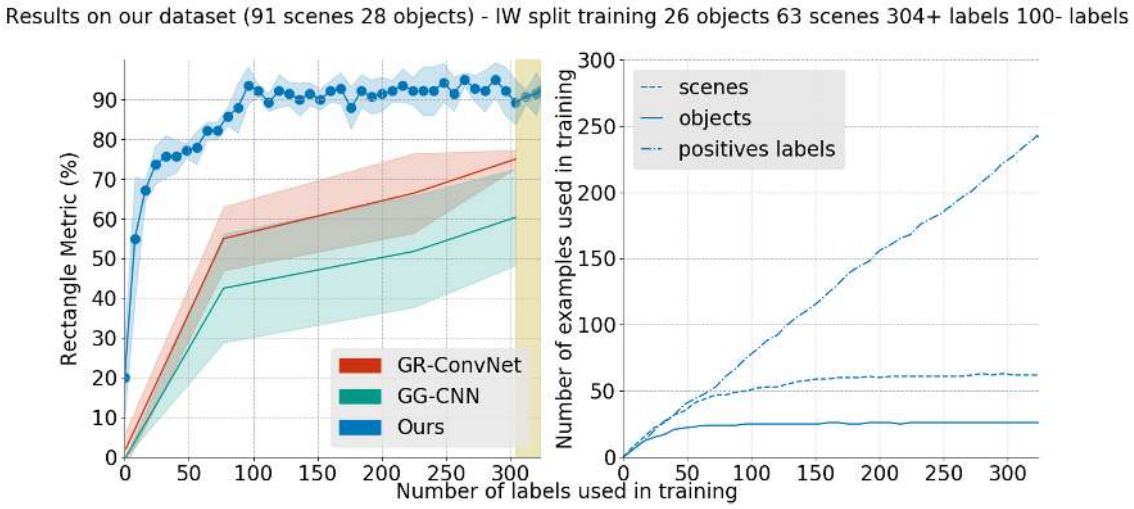


FIGURE 3.11 – Métrique du rectangle pour notre jeu de données. L'axe des abscisses correspond au nombre d'étiquettes utilisées pour l'entraînement (données d'entraînement et de validation, 641 scènes) et l'axe des ordonnées au pourcentage de la métrique rectangle par rapport au nombre de scènes dans l'ensemble de test (244 scènes). La métrique rectangle compte combien de fois la saisie sélectionnée correspond à au moins l'une des étiquettes positives. Pour l'ensemble de données Cornell, le graphique du haut montre la métrique rectangle de 0 à toutes les étiquettes disponibles (GR-ConvNet et GG-CNN n'utilisent pas les étiquettes négatives, donc ils ne peuvent pas utiliser toutes les étiquettes), le graphique en bas à gauche correspond aux mêmes données mais se concentre sur les résultats de 0 à 400 étiquettes, et le graphique en bas à droite indique le nombre de scènes et d'objets par rapport au nombre d'étiquettes utilisées pour l'entraînement. Pour mieux comprendre les performances de notre pipeline, nous comparons deux "ablations" : remplacer le classificateur GP par un modèle de régression logistique pour la classification binaire ("Ours-LR") et utiliser un classificateur binaire CNN au lieu de la combinaison VAE+GP ("Ours-CNN").

FIGURE 3.12 – Nous avons sélectionné ces objets selon plusieurs critères. Ils doivent présenter de multiples configurations de pose stables pour pouvoir être disposés sur une table. Ils doivent également permettre des prises distinctives et spécifiques, par exemple les outils ou les couverts qui doivent être saisis par le manche, les jouets par certaines parties, les bouteilles par leur bouchon, etc... De plus, nous avons veillé à choisir une variété d'objets de différentes tailles, formes et couleurs. Enfin, certains de ces objets présentent des caractéristiques particulières, comme le fait d'être transparents, déformables ou articulés.

rectangle (Sec. 3.1). Nous n'avons pas pu exécuter Dex-Net 4.0 [113] sur le jeu de données Cornell car les objets sont trop éloignés de la caméra (Dex-Net requiert 0,5 à 0,7m) et le jeu de données ne donne pas les paramètres intrinsèques de la caméra. GG-CNN et GR-Convnet sont deux algorithmes récents qui utilisent un réseau neuronal entièrement révolutionnaire pour générer la qualité de la saisie et les poses/largeurs de saisie (x, y, θ, w) à chaque pixel des scènes RGB-D et pour échantillonner les candidats à la saisie. GR-ConvNet affiche une exactitude de pointe de 97,7% sur le jeu de données Cornell. Pour évaluer la contribution du GP, nous avons également comparé à une base de référence dans laquelle une régression logistique (de Scikit-Learn) est utilisée à la place du GP.

Pour les expériences suivantes, nous nous intéressons à la comparaison des performances de notre méthode (en termes de suivi des préférences) en fonction du nombre de labels disponibles. Pour diviser notre jeu de données par image, nous classons les scènes RGB-D par objet et choisissons aléatoirement une scène par objet pour la base de données de test, sur lequel chaque algorithme est testé.

Nous générons les base de données d'entraînement et de validation pour un nombre spécifique de prises étiquetées en sélectionnant aléatoirement des points de données en dehors de la base de données de test. Ces étiquettes sont ensuite divisées en 5 groupes afin d'effectuer une validation croisée 5 fois pour entraîner GG-CNN et GR-ConvNet. Chaque

Dataset	# of scenes	with single objects	with objects in heaps	# of objects	# of VAE grasps
Ours	782 - 1280x720	369	413	124	334286
Cornell	885 - 640x480	885	0	244	43644

TABLE 3.1 – Jeux de données utilisés pour l'entraînement VAE.

méthode est entraînée avec les mêmes étiquettes disponibles pour l'entraînement, avec cette différence : GG-CNN et GR-ConvNet rejettent les saisies négatives pour l'entraînement (dans ces méthodes, tout ce qui n'est pas positif est considéré comme négatif), et notre méthode rejette les données de validation (puisque nous n'en avons pas besoin pour le GP). Par exemple, pour 100 étiquettes sélectionnées au hasard (par exemple, 60 positives et 40 négatives), chaque pli contient 80 étiquettes pour l'entraînement et 20 pour la validation ; notre méthode est formée avec 80 étiquettes alors que GG-CNN et GR-ConvNet utiliseraient en moyenne 48 étiquettes (36 étiquettes pour l'entraînement et 12 étiquettes pour la validation). Veuillez noter que nous rapportons les données en utilisant le nombre d'étiquettes utilisées (entraînement et validation), et non le nombre d'étiquettes choisies au hasard, afin que la comparaison soit aussi équitable que possible.

Nous exécutons notre méthode 5 fois pour chaque nombre d'étiquettes (chaque exécution est indépendante) ; nous testons avec $0, 10, 20, \cdots, 500$ étiquettes et $1000, 2000, \cdots, 4558$ étiquettes. Globalement, nous lançons donc 280 ($56 \times 5 = 280$) exécutions indépendantes de notre algorithme d'apprentissage (et donc 280 tests). Pour GR-ConvNet et GG-CNN, nous testons avec des étiquettes de $0, 100, 300, 400, 1000, 3000$, et 3588 (ces méthodes n'utilisent que les étiquettes positives) et 5 répétitions (total de 45 pour chaque ligne de base). Nous entraînons la partie VAE de notre méthode une fois pour chaque base de données $\mathcal{D}$, en utilisant une répartition de 70%-30% pour l'entraînement et la validation. Le réseau VAE est entraîné pendant 48 heures sur une Nvidia GTX 1080 Ti en utilisant le cadre Keras et l'optimiseur Adam avec les paramètres suivants : $|B| = 64, \eta = 0.001, w = 128, h = 128, \beta = 1.0, m = 32$. GG-CNN et GR-ConvNet sont tous deux entraînés sur Nvidia GTX 1080 avec les paramètres fournis et des augmentations de données adaptées au Cornell Dataset.

Nous avons constitué un jeu de données $\mathcal{D}$ d'images RGBVB-D d'objets vu de haut pour entraîner la β-VAE. Les images sont obtenues avec une camera Intel RealSense D415 Depth 65 centimètres au dessus des objets, ceux ci provenant de diverses expériences robotiques et de la base de données YCB [28].

3.4.1 Résultats qualitatifs

Nous avons d'abord vérifié que notre méthode correspond à nos attentes sur un objet dont le choix est clair (Fig. 3.1). Dans cette expérience, un marteau (jouet) doit être pris par le haut du manche (près de la tête), et non par le bas (loin de la tête). Nous avons choisi cet exemple parce qu'un marteau a une distribution de masse particulièrement non-uniforme qui ne peut pas être déduite de la forme seule : les prises qui ne sont pas proches de la tête ont peu de chances de réussir.

Pour cette expérience préliminaire, nous avons utilisé une seule image avec deux étiquettes : une étiquette positive pour la partie supérieure du manche et une étiquette négative pour la partie inférieure (Fig. 3.1). Nous avons ensuite capturé 21 scènes sup-

plémentaires du même marteau dans différentes positions et évalué combien de fois notre méthode a choisi de saisir le marteau par la partie supérieure (pour effectuer cette évaluation, nous avons étiqueté les 21 scènes, mais n'avons pas utilisé ces étiquettes lors de l'entraînement).

Les résultats montrent que notre méthode sélectionne la bonne prise dans 19/21 scènes. (Fig. 3.13) et n'échoue que dans les 2 scènes où le marteau est posé à la verticale sur la table (pour lesquelles le manche n'est pas accessible avec une prise de haut en bas).

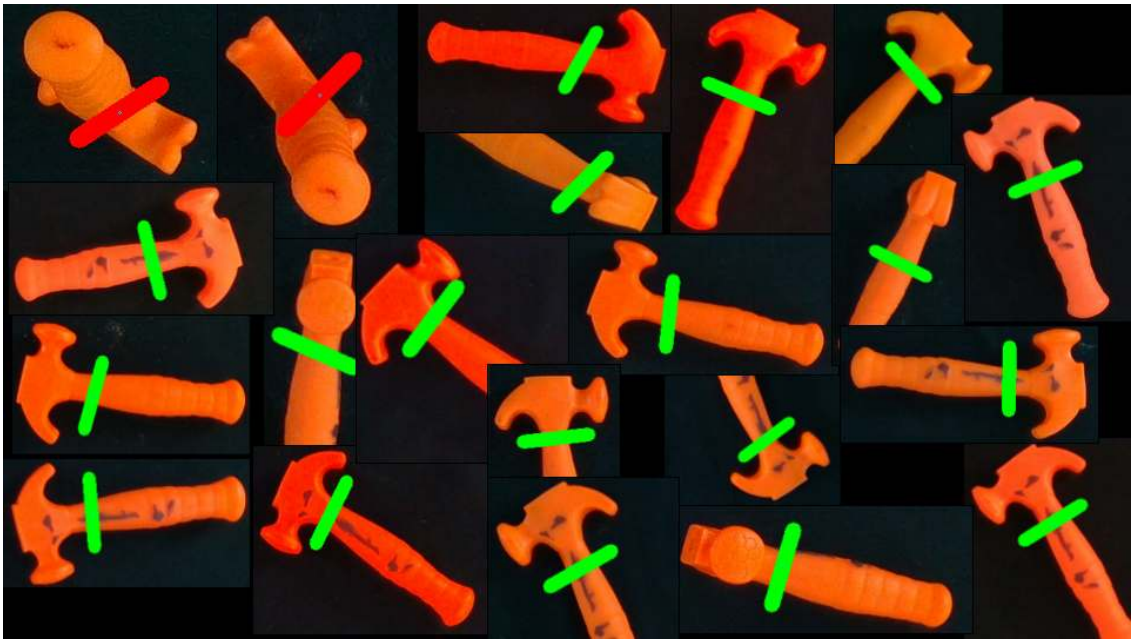


FIGURE 3.13 – Saisies prédites sur les 21 scènes après entraînement avec deux étiquettes.

3.4.2 Résultats quantitatif

Après environ 16 scènes d'exemple tirées du jeu de données Cornell (16 étiquettes aléatoires, qui correspondent à environ 13 exemples positifs pour environ 14 objets), notre méthode sélectionne la bonne saisie (selon la métrique du rectangle) pour 66% des scènes de test (244 scènes) (Fig. 3.10) qui comprennent de nombreux objets non vus (244 objets au total, alors qu'au mieux 16 objets différents ont été vus). À titre de comparaison, la sélection aléatoire de la saisie parmi celles générées conduit à un score de 29% (cela correspond à la valeur "0 étiquette" sur le graphique), ce qui signifie que le classificateur GP apprend des connaissances utiles. Après 68 étiquettes, notre méthode prédit avec exactitude les saisies pour 76% des scènes, alors que GR-ConvNet sélectionne la bonne saisie dans seulement 55% des scènes testées, et GG-ConvNet seulement 49%. Notre méthode dépasse 80% avec 80 étiquettes (63 objets différents, 71 scènes, 54 étiquettes positives) et 89,5% avec 344 étiquettes. Le meilleur score atteint par GR-ConvNet après 3588 étiquettes [5] est de 86.8% et par GG-CNN [120] de 77.9%. Dans l'ensemble, notre méthode

5. Veuillez noter que les auteurs de GR-ConvNet rapportent une exactitude de 97.7% avec toutes les étiquettes disponibles, mais nous n'avons pas pu obtenir le même score avec leur code (ils ne rapportent

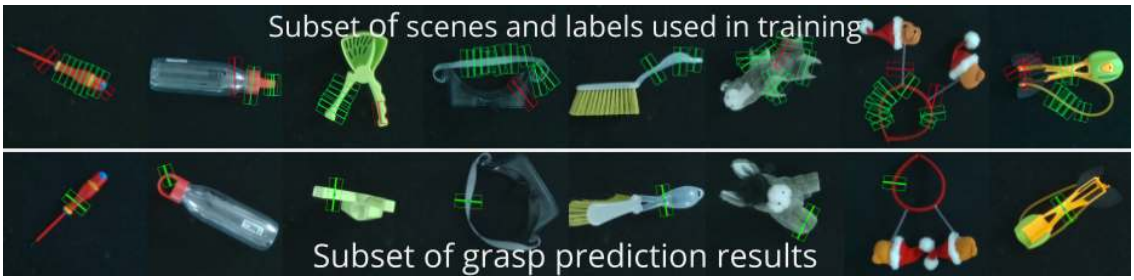


FIGURE 3.14 – Résultats typiques de notre jeu de données : la ligne supérieure montre les objets dans les scènes utilisées pour l'entraînement (d'autres scènes de ces objets ont également été utilisées), et la ligne inférieure montre certaines de nos saisies prédites sur des scènes non vues du même objet. Pour évaluer l'efficacité des données, un sous-ensemble aléatoire d'étiquettes est utilisé (par exemple, seulement 2 étiquettes pour chaque objet). Les prédictions suivent les règles arbitraires de l'expert pour chaque objet : une partie de l'objet est autorisée (le manche du tournevis, le bouchon de la bouteille rouge, la ficelle des lunettes, ...) et une autre ne l'est pas (les poils de la brosse, la tête de la peluche de l'âne, la nageoire de la torpille, ...).

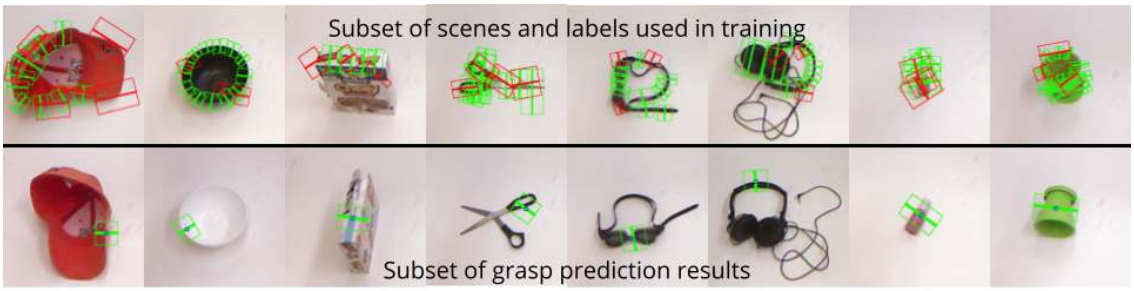


FIGURE 3.15 – La ligne supérieure montre les objets dans les scènes disponibles pour l'entraînement, et la ligne inférieure montre certaines des saisies prédites sur des scènes non vues du même objet.

surpasse clairement le meilleur score de toutes les lignes de base après seulement 344 étiquettes (contre plus de 3000 pour les lignes de base), et quelques centaines d'étiquettes suffisent pour obtenir plus de 80% des saisies correctes.

Les résultats sont similaires pour notre base de données (Fig. 3.14), qui est plus petite mais similaire à la base de données Cornell (objet unique avec étiquettes) notre méthode atteint un score de 74% avec seulement 24 étiquettes (provenant de 22 scènes, 15 objets différents, dont 20 sont positifs), et 93.57% avec 96 étiquettes. Pour fournir un point de référence, avec 300 étiquettes, GR-ConvNet obtient un score de 75% et GG-CNN un score de 60%.

aucun score avec un sous-ensemble de la base de données Cornell, comme nous le faisons ici). Nos résultats sont cohérents avec les résultats rapportés dans la littérature GG-CNN [120].

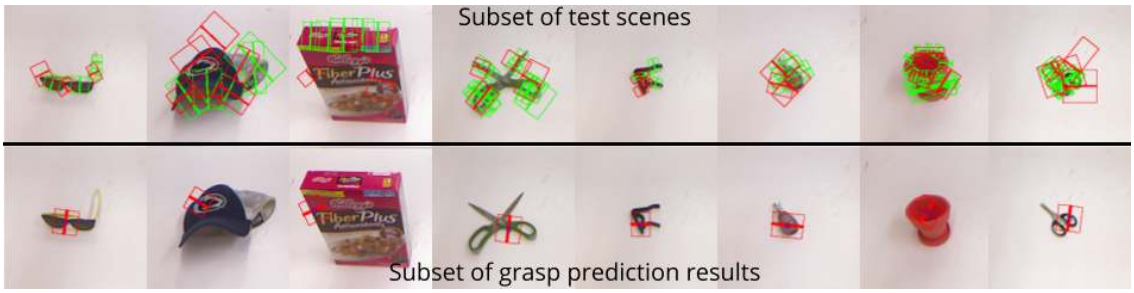


FIGURE 3.16 – Résultats typiques sur la base de données Cornell, la ligne supérieure montre les objets dans les scènes utilisées pour l'entraînement (des scènes supplémentaires de ces objets ont également été utilisées). La deuxième ligne montre les saisies prédites qui ne satisfont pas la métrique du rectangle. Ces 8 scènes de la base de données Cornell échouent même après avoir été entraînées avec des étiquettes de ces objets provenant d'autres scènes.

3.4.3 Étude d'ablation

Notre pipeline combine un VAE pour apprendre un espace latent générique avec un GP pour classifier les candidats à la saisie dans cet espace latent. Pour comprendre la contribution du GP, nous l'avons remplacé par un classificateur de régression logistique (de Scikit-Learn). Les résultats (Fig. 3.10, "Ours-LR") montrent que la régression logistique conduit à des scores similaires à ceux obtenus par GR-ConvNet, ce qui montre que le classificateur GP est un composant clé. Pour évaluer la contribution du VAE entraîné sur de nombreuses images non étiquetées, nous avons ensuite entraîné un classificateur CNN avec la même architecture que le VAE (il prend les mêmes patches en entrée) mais qui sort directement la classe du candidat à la saisie (correcte ou incorrecte). Ce CNN est entraîné uniquement sur les patches étiquetés. Les résultats montrent (Fig. 3.10, "Ours-CNN") que cette approche simple peut conduire à des résultats compétitifs mais est clairement surpassée par notre pipeline. Dans l'ensemble, l'entraînement sur des images non étiquetées et l'utilisation d'un GP sont nécessaires pour obtenir un bon score selon la métrique du rectangle.

3.4.4 Testes avec un robot Franka-Emika Panda

La vidéo[6] montre un robot Franka-Emika Panda qui effectue les saisies apprises à l'aide de notre pipeline et de notre base de données (Fig. 3.1). Pour chacun des 28 objets (Fig. 3.12), le robot est positionnée à 65 cm au-dessus de la table, notre pipeline sélectionne la saisie et le robot l'exécute en utilisant les algorithmes de planification de la bibliothèque MoveIt [36].

L'objectif de notre méthode est de suivre les préférences de l'expert, et non de réaliser les saisies les plus réussies : l'expert peut exiger une saisie beaucoup plus difficile que les autres pour une raison qui lui est propre (c'est ce qu'évalue la métrique du rectangle). Dans ces expériences sur les robots, 17% des saisies ne correspondaient pas à ce que l'expert attendait et 20% des saisies étaient infructueuses (ces deux ensembles d'échecs ne

6. Vidéo disponible à l'adresse `https://youtu.be/dJ1fkcught4`

se chevauchent pas complètement). La plupart des échecs sont dus à des problèmes liés à l'étape de génération des saisies (par exemple, la saisie préférée n'a pas été proposée par le générateur ou aucune bonne saisie n'a été proposée).

3.4.5 Avec classification d'objet

Dans certains cas, il est essentiel de reconnaître les objets que l'on manipule. Par exemple, lors du traitement des déchets radioactifs, il est nécessaire de trier les objets en fonction de leur type. Ces déchets sont composés d'objets courants tels que des bouteilles en plastique et des tissus, mais aussi d'une proportion importante d'objets industriels tels que des blocs de bois, des boîtes métalliques, des chaînes, des gants, des tuyaux et autres objets métalliques. L'identification et la classification de ces objets est cruciale dans le traitement de ces déchets. Dans cette section, nous évaluons plusieurs améliorations possible de notre pipeline pour effectuer cette reconnaissance.

Dans le cas de la reconnaissance d'objets saisis, la base de données Cornell étiquette les objets présents dans chaque scène (244 objets différents, 231 avec plusieurs scènes). De nombreux objets sont similaires, par exemple, chaque paire de lunettes de couleur différente est considérée comme un objet distinct. La plupart des objets ont 4 scènes, certains en ont 3, 2 ou 1, et quelques-uns en ont 5, ce qui est peut et déséquilibré (voir Fig. 3.17).

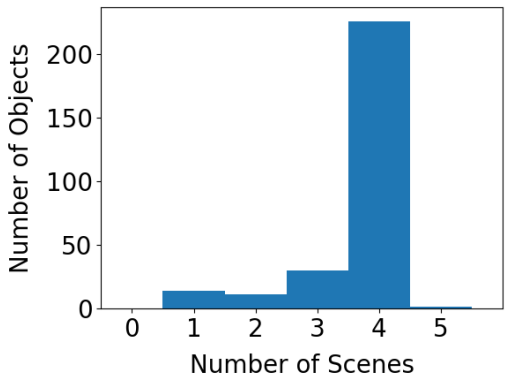


FIGURE 3.17 – Distribution du nombres de scènes par objets de la base de données Cornell.

Une première tentative, qui n'a pas donné de bons résultats, consistait à remplacer le classificateur binaire par un classificateur multi-classes SVM (à noyau quadratique rationnel) et à utiliser de l'augmentation de données (Fig. 3.18) sur les données d'entraînement. On obtient une précision de 9,6% pour les étiquettes et 5,08% pour les candidats de ces scènes. Si l'on considère le top 5 des classes prédites pour chaque patch, la précision monte à 23,58% pour les étiquettes et 15,74% pour les candidats de ces scènes.

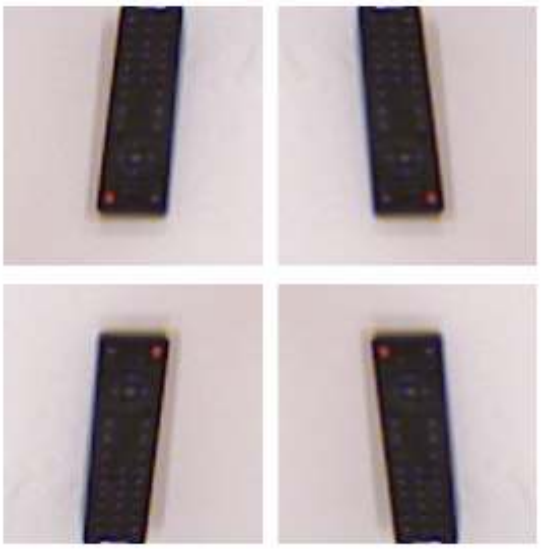


FIGURE 3.18 – Retourner verticalement (haut-bas) ou horizontalement (gauche-droite) l'image conserve l'information de la prise de vue si la caméra est parallèle à la table.

Les autoencodeurs variationnels (VAE) sont des modèles génératifs qui apprennent une représentation latente des données d'entrée. L'objectif principal des VAE est de générer de nouveaux échantillons de données similaires aux données d'apprentissage, et ils ne sont pas conçus explicitement pour des tâches de classification. La représentation dans l'espace latent (une distribution gaussienne isotrope dense) ne semble pas être adéquate pour cette tâche. Comme nous disposons de l'encodeur du VAE, nous pouvons tester d'entraîner un classificateur par transfert (voir sous-sous-section 2.5.3.1). Pour cela, nous figeons toutes les couches convolutives de l'encodeur (qui ont déjà appris des caractéristiques utiles à partir des données d'entraînement originales) afin d'éviter de détruire les informations qu'elles contiennent et ajoutons des couches au modèle afin de réaliser la classification. Les couches ajoutées sont les suivantes :

- GlobalAveragePooling2D : cette couche calcule la moyenne de la carte des caractéristiques pour chaque canal, ce qui donne une sortie de taille fixe indépendamment de la taille d'entrée. Elle réduit les dimensions spatiales de l'entrée et fournit un résumé des caractéristiques les plus importantes de l'entrée.
- BatchNormalization : cette couche normalise les sorties de la couche précédente en les ajustant et en les mettant à l'échelle. Elle aide à stabiliser le processus d'apprentissage et accélère l'entraînement.
- Dense : cette couche entièrement connectée prend la sortie de la couche précédente et applique une transformation linéaire. Cette couche est ajoutée pour apprendre des représentations complexes de l'entrée et contribue à améliorer les performances du modèle (fonction d'activation ReLU).
- Dropout : cette couche supprime de manière aléatoire une fraction des unités d'entrée, ce qui contribue à prévenir le surapprentissage. Il s'agit d'une technique de régularisation qui améliore la généralisation du modèle.
- Dense : il s'agit de la dernière couche de sortie du modèle. Elle comporte un nombre d'unités correspondant au nombre de classes. La couche applique une fonction

d'activation softmax, qui convertit la sortie de la couche en une distribution de probabilités sur les classes.

En utilisant cette approche, nous obtenons de bien meilleurs scores comparés à ceux du classificateur multi-classes SVM, soit 58,52% pour les étiquettes et 59,6% des candidats de ces scènes. Si on prend en compte le top 5 des classes prédites, les scores augmentent à 85,2% et 85,67%. En gardant la même architecture mais en remplaçant les couches venant de l'encodeur par celles pré-entraînées sur ImageNet (une grande base de données contenant plus de 14 millions d'images, les architectures classiques dont s'inspire notre VAE proposent souvent des poids pré-entraînés sur cette base de données) [47] (dans ce cas, nous n'utilisons plus que les données RGB), les scores augmentent à 71,46% pour les étiquettes et 71,92% des candidats de ces scènes. Ce résultat n'est pas surprenant car les poids pré-entraînés ont été appris à partir d'une base de données beaucoup plus grande et plus diversifiée que le VAE, malgré l'utilisation des données de profondeurs.

3.4.6 Avec segmentation

Dans le cas où la segmentation des scènes est fournie dans la base d'apprentissage, cette information peut être optionnellement apprise. En intégrant le masque de segmentation des patches comme une sortie supplémentaire du VAE (qui reconstruit les données RGB-D et fournira donc également une estimation de la segmentation), nous avons pu obtenir des résultats de segmentation légèrement améliorés. Les résultats de l'évaluation montrent un score médian IoU de 0,878 et un score moyen IoU de 0,854 avec un temps de calcul moyen de 0,78 seconde par scène (contre un score médian de 0.8777 et un score moyen IoU de 0,83 pour un temps de calcul moyen par scène de 0,53 seconde pour la méllieur combinaison de segmentation obtenu précédemment, voir sous-section A.6.1).

Ces résultats sont probablement moins performants qu'une méthode de segmentation plus spécialisée telle que U-Net [150], ils montrent que la combinaison d'un VAE et d'un masque de segmentation peut permettre d'obtenir une segmentation précise sans changer le reste du pipeline de préhension. En effet, si les masques de segmentation sont disponibles pour entraîner le VAE, il peut être facilement intégré dans un pipeline existant de préhension. Cette approche est donc une option intéressante pour les applications de robotique où les masques de segmentation sont disponibles et où il est souhaitable de maintenir un pipeline simple et facile à implémenter.

3.5 Conclusions

Notre pipeline est très économe en données car il utilise un générateur de saisie basé sur la forme comme préalable, au lieu d'apprendre les saisies uniquement à partir des étiquettes. Une conséquence directe est que notre pipeline est au moins aussi bon que le générateur de saisie utilisé : il choisit toujours parmi les saisies proposées. Lorsque des générateurs de préhension plus efficaces seront développés, ils pourront être directement utilisés pour créer l'espace latent et les candidats, et notre pipeline deviendra plus efficace.

Ce chapitre se concentre sur l'entraînement hors ligne afin de le comparer à l'état de l'art sur des bases de données bien définis. Cependant, notre pipeline s'adapterait bien à des scénarios en ligne dans lesquels un "superviseur" ne corrige le robot que lorsqu'il se

trompe, c'est-à-dire lorsque le générateur de prise basé sur la forme est erroné. Le superviseur agirait alors comme un professeur avec un élève compétent, avec une supervision minimale. Dans le chapitre 6, nous avons mis en œuvre une configuration "human-in-the-loop" avec des générateurs de préhension récents afin d'évaluer les performances de ce genre de processus d'apprentissage en ligne.

Chapitre 4

Apprentissage de la hauteur pour les saisies de haut en bas avec le capteur DIGIT

Ce chapitre reprend et traduit la publication :
(IEEE ICRA 2023) - Learning height for top-down grasps with the DIGIT sensor
Thais Bernardi*, **Yoann Fleytoux***, Jean-Baptiste Mouret, Serena Ivaldi

Thais Bernardi (Stagiaire M2) a participé à la conception, à la réalisation et à l'analyse des expériences de caractérisation du capteur, a écrit le logiciel correspondant à la captation des données de celui-ci, et a préparé et révisé l'article. Elle a été co-supervisée par moi et mes encadrants.

J'ai participé à la conception et à la réalisation des expériences de caractérisation du capteur, j'ai conçu et réalisé les expériences de collecte et d'apprentissage de la hauteur des saisies, écrit le logiciel correspondant, et préparé et révisé l'article.

Jean-Baptiste Mouret a formulé le problème, supervisé et contribué à la conception expérimentale du travail de recherche, à la discussion et à l'interprétation des résultats, ainsi qu'à la rédaction et à la révision de l'article.

Serena Ivaldi a formulé le problème, supervisé et contribué à la conception expérimentale du travail de recherche, à la discussion et à l'interprétation des résultats, ainsi qu'à la rédaction et à la révision de l'article.

Nous avons examiné, dans le chapitre 3, divers générateurs de préhension fondés sur l'image tels que Dex-Net [108], GG-CNN [120], et GR-ConvNet [89]. Ces systèmes permettent d'identifier des préhensions en 4D, aussi désignées sous le terme de *top-grasp*.

* Les deux premiers auteurs de l'article scientifique en question ont contribué à parts égales à la recherche et à la rédaction de l'article.

Une préhension 4D se définit couramment par la fonction (x, y, z, θ), où (x, y, z) représentent les coordonnées cartésiennes du centre de préhension et θ désigne l'orientation du préhenseur sur le plan (voir sous-section 2.1.4). Toutefois, si les composantes x et y du centre de saisie sont bien établies, la hauteur de préhension z est généralement obtenue par une méthode heuristique fondée sur la hauteur maximale déterminée par le nuage de points, comme illustré dans le chapitre 3.

Des représentations de préhension largement utilisées, telles que les rectangles orientés [75, 49] ou les cartes de préhension au niveau du pixel [4, 58, 186, 188] prennent en compte le centre de la prise, la distance entre deux mâchoires, la taille de la pince et son orientation, mais ne codent pas la hauteur de prise z. Les heuristiques pour la hauteur fonctionnent la plupart du temps, surtout lorsque les objets sont épais ou pleins, mais elles sont souvent inadéquates lorsque les objets ont des parties fines, des parties transparentes, des trous ou des formes courbes, en raison de l'erreur intrinsèque dans l'estimation de la hauteur à partir du nuage de points ou simplement de l'absence de connaissance sur ce qui se trouve derrière la surface de l'objet, ce qui nécessite le modèle 3D de l'objet. La figure 4.1 montre un exemple d'échec de saisie dans ce sens : deux générateurs de saisie de l'état de l'art, Dex-Net [108] et GPD [136], échouent à cause de la hauteur de saisie (dans un cas trop basse, dans l'autre trop haute, dans les deux cas la pince ne touche pas correctement l'objet lors de la fermeture).

Dans ce chapitre, nous abordons ce problème en proposant de prédire la meilleure hauteur de saisie à partir de l'image RGB-D, en utilisant un régresseur préalablement entraîné avec une base de données de la meilleure hauteur de saisie parmi plusieurs candidats de saisie. Nous postulons que la "meilleure hauteur de saisie" peut être apprise automatiquement à partir d'une collecte de données pilotée par capteur, où le robot tente différentes hauteurs candidates $(z_0, z_1, \dots)$ sur le même candidat de saisie 3D (x, y, θ), en utilisant les informations de contact fournies par un capteur tactile. Dans notre travail, nous avons utilisé DIGIT [92], un capteur tactile optique qui fournit une mesure de l'ellipsoïde de contact, qui s'est avéré expérimentalement être un bon prédicteur de la stabilité et du succès de la saisie. Nos expériences avec plusieurs objets montrent que la hauteur de saisie prédite améliore légèrement le succès de la saisie, de 6% globalement sur tous les objets, mais surtout, elle permet de saisir avec succès des objets qui autrement échouaient, avec une amélioration allant jusqu'à 40%.

Les principales contributions de ce chapitre sont : i) la caractérisation du capteur DIGIT, avec plusieurs expériences visant à déterminer si sa sortie peut être utilisée pour la prédiction de la stabilité de la saisie ; ii) la validation expérimentale de l'ellipsoïde de contact optique en tant que prédicteur de la stabilité et de la réussite de la saisie ; iii) la méthode permettant d'identifier le meilleur candidat à la saisie à l'aide des mesures de DIGIT, puis d'entraîner un régresseur de hauteur de saisie basé sur des patchs de saisie latents. Tous nos résultats sont le fruit de plusieurs expériences réelles avec le robot Franka équipé d'un préhenseur standard, sur lequel sont montés deux capteurs Digit (acquis auprès de GelSight).

FIGURE 4.1 – Problème typique des saisies planes : la hauteur du centre de saisie est souvent fixée par une heuristique basée sur la surface la plus haute. Si la forme de l'objet est inconnue, cela peut conduire à des saisies infructueuses. À gauche, une image en échelle de gris d'un objet générée à partir du nuage de points capturé par la caméra RGB-D montée sur l'effecteur du robot. Au centre et à droite, deux saisies descendantes générées par Dex-Net [108] et GPD [136], deux générateurs de saisie classiques. La prise de Dex-Net au centre est trop basse, l'objet n'est pas saisi par les mâchoires de la pince, tandis que la prise de GPD à droite est trop haute pour être réussie (en raison de la forme arrondie du manche, l'objet glisse et est poussé vers le bas lors de la fermeture de la pince). Sans connaissance préalable de l'objet et de son épaisseur, une mauvaise prédiction de la hauteur peut conduire à l'échec.

4.1 Caractérisation du capteur DIGIT

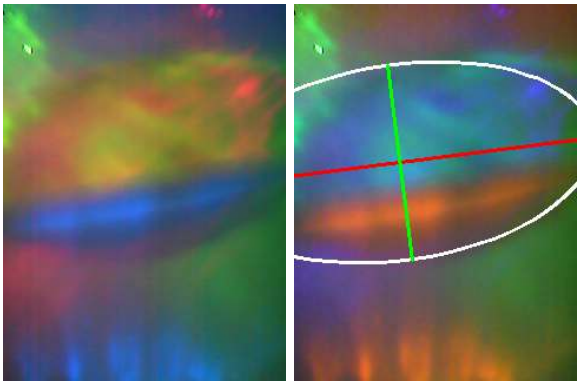
DIGIT [92] est un capteur tactile basé sur la vision : il se compose d'une fenêtre en acrylique qui est, du côté le plus externe, recouverte d'un élastomère et, du côté interne, éclairée par trois LED de couleur (rouge, vert et bleu). Une caméra CMOS, à l'intérieur, capture la fenêtre en acrylique sous forme d'image, qui constitue la sortie du capteur.

Lorsqu'un objet appuie sur la surface du capteur, la fenêtre en acrylique se déforme et l'effet de l'objet est perceptible dans l'image de sortie.

Tous les capteurs DIGIT utilisés dans ce travail ont utilisé l'élastomère réfléchissant ; ils étaient réglés sur les configurations par défaut (acquisition d'image à 60 *fps*, intensité d'éclairage au maximum), sauf indication contraire. Des exemples de sorties du capteur pour différents objets sont présentés dans la figure A.10 et dans la vidéo[7]. Les objets à surface plate sont les moins perçus par DIGIT et ne sont pas évidents à reconnaître à l'œil nu dans l'image de sortie (Fig. A.10(d)), contrairement aux petits objets ou aux objets avec des arêtes (a et b). Les objets courbes, lorsqu'ils sont saisis ailleurs qu'au centre, sont également moins perceptibles (c).

Pour quantifier la réponse DIGIT de tout contact avec le capteur, nous considérons

7. Vidéo disponible à l'adresse `https://youtu.be/aZ1Hjaziv6Y`



(a) (b)

FIGURE 4.2 – (a) Réponse de DIGIT en présence d'un contact avec un objet ; (b) l'ellipsoïde ϵ entourant la zone de contact, calculée par *PyTouch*.

deux métriques. La première est δP, c'est-à-dire la somme multicanal (puisque le capteur a des canaux RGB) de la différence pixel par pixel entre la sortie d'image actuelle P_{image} et une image de référence $P_{no-contact}$, obtenue lorsqu'il n'y a pas de contact :

$\delta P = \frac{1}{n \times m} \sum_{i=0}^{n} \sum_{j=0}^{m} (p_{i,j_{image}} - p_{i,j_{no-contact}})$,

où p est la valeur de chaque pixel dans la plage $[0, 255]$ et les images sont de taille $n \times m$. La seconde est la surface de contact de l'objet avec le capteur. La surface de contact est calculée par la fonction *ContactArea* de la bibliothèque *PyTouch* [93], qui trouve l'ellipse englobant la surface de contact ; comme le montre la figure 4.2, ϵ, la surface de contact approximée par l'ellipsoïde peut être calculée en pixels.

Afin de caractériser le comportement du capteur, nous avons effectué les tests suivants :

Test 1 : dérive. L'objectif est de déterminer s'il y a une dérive sur la mesure de base, c'est-à-dire celle lorsque le capteur n'est pas en contact avec un objet, afin de prévoir des procédures de re-calibration ou de réinitialisation adaptées si nécessaire. Le test consiste à collecter des données à une fréquence de 1 Hz en deux sessions distinctes de 30 et 90 minutes, sans aucun contact.

Test 2 : sensibilité à la lumière ambiante. Pour vérifier si l'éclairage de l'environnement modifie la sortie du capteur, nous collectons des données à une fréquence de 1 Hz lors d'une session de 2 minutes, avec trois niveaux différents d'éclairage environnemental générés artificiellement par une lampe et fixés de manière aléatoire.

Test 3 : relation entre la sortie du capteur et la force de contact. L'objectif est de déterminer si la sortie du DIGIT est en relation avec la force de contact. Pour simplifier, nous ne réalisons le test que pour la force normale. Pour réaliser le test, nous utilisons la pince Franka, en montant des DIGIT sur les deux «doigts», et un capteur Optoforce (modèle OMD-20-FG-100N) pour mesurer la force de contact. La première partie du test consiste à fermer la pince avec différents réglages de force et de vélocité, en utilisant l'API libfranka. Puis, la seconde consiste à fermer la pince pour saisir une épingle (celle de la Fig. A.10 (a)) de 0.005m de diamètre et 0.015m de hauteur, positionnée verticalement, son bord proche du milieu du capteur. Ce type d'objet présente des bords bien définis qui sont facilement détectés par le DIGIT.

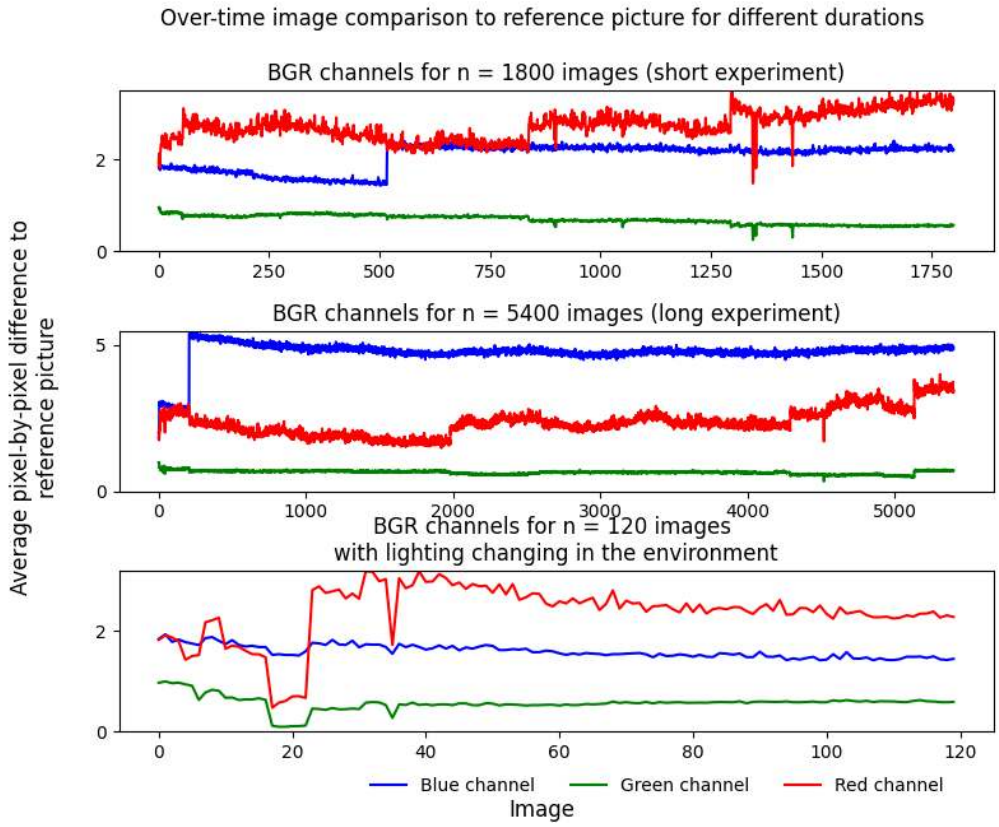


FIGURE 4.3 – Test 1 & 2 : sortie du DIGIT en tant que δP pour tester la répétabilité dans le temps pour différentes sessions, et pour des conditions d'éclairage de l'environnement changeant.

Les commandes d'ouverture et de fermeture de la pince sont envoyées pendant plusieurs minutes, en boucle, avec différentes forces de réglage. Les images du DIGIT sont acquises à 60Hz.

Test 4 : répétabilité des mesures de contact. L'objectif est de vérifier si les images ont des mesures cohérentes en présence du même contact, de la même force, afin de détecter une éventuelle hystérésis ou dérive. Le test est exécuté avec la pince équipée de deux DIGITs comme "doigts". Il consiste à fermer et ouvrir la pince 180 fois, en appliquant la même force, avec un objet placé entre les 2 DIGIT, en collectant des images à 60Hz.

4.1.1 Résultats de la caractérisation

Nous rapportons ici les résultats des 4 tests visant à caractériser le comportement de DIGIT. La Fig. 4.3 montre les résultats du **Test 1 & 2**. Chaque canal de couleur a des valeurs dans la plage [0,255]. Dans les deux premiers graphiques (session courte et longue), les valeurs de chaque canal sont presque constantes et proches de 0, ce qui est cohérent avec l'absence de contact (bien qu'il y ait quelques erreurs de flux d'images visibles qui provoquent des pics). Le troisième graphique montre que l'éclair n'influence pas la mesure de base.

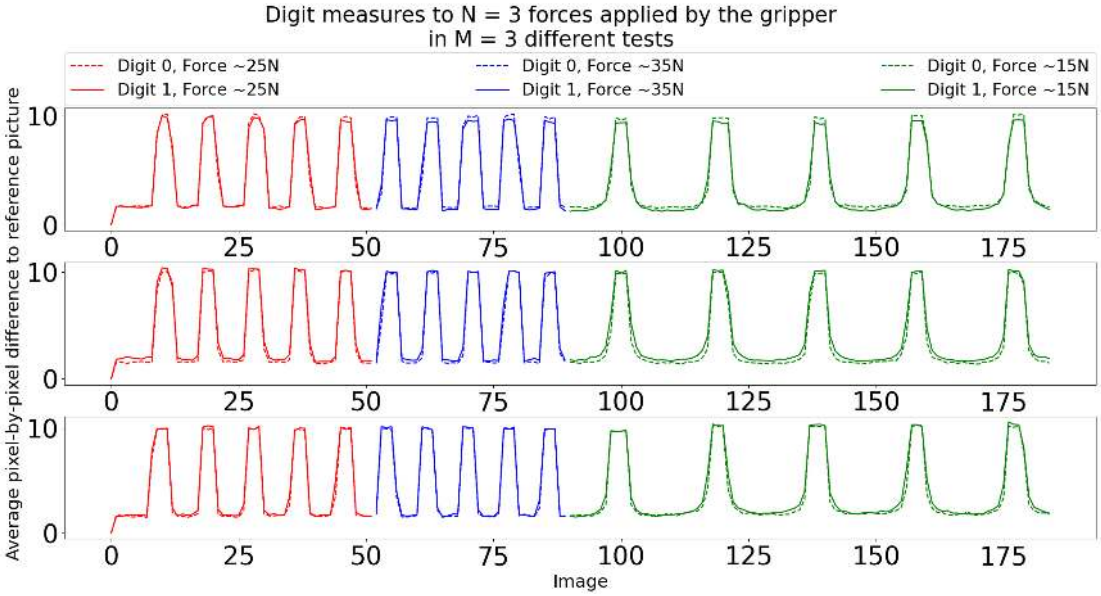


FIGURE 4.4 – Test 3 : sortie de DIGIT sous forme de δP en fonction de différentes forces de contact.

La Fig. 4.4 montre les résultats du **Test 3**, où 3 ensembles de forces ont été appliqués aux deux DIGIT. Nous utilisons la métrique δP (Sec. 4.1), c'est-à-dire la différence pixel par pixel moyennée sur les 3 canaux. L'objet de la broche a été choisi de telle sorte que la surface de contact ϵ ne change pas pendant l'expérience. Les résultats des différentes répétitions sont les mêmes : il n'y a pas de relation entre la sortie du DIGIT en termes de valeurs de pixel et la force de contact. Cela signifie que **les mesures pixel par pixel avec DIGIT ne peuvent pas être utilisées pour distinguer les forces de contact, mais seulement les surfaces de contact**.

Enfin, la Fig. 4.5 montre le résultat du **Test 4**. Nous avons exécuté des saisies périodiques avec la pince, en appliquant la même force sur la broche, 180 fois. Deux choses sont à noter. Premièrement, la valeur de δP pour les deux doigts est différente : cela pourrait être attribué à une distribution asymétrique de la force de contact, même si nous avons utilisé un objet symétrique pour éviter ce problème. Deuxièmement, au fur et à mesure que le temps avance, nous observons d'étranges «sauts» consécutifs dans la mesure basée sur les pixels, apparaissant à des moments différents pour les deux capteurs. Nous avons réalisé la même expérience plusieurs fois, et avons toujours observé un comportement similaire, bien que les «sauts» semblent se produire de manière aléatoire. Cette expérience suggère que la réponse des pixels de DIGIT aux mêmes forces de contact n'est pas constante dans le temps, mais qu'elle est soumise à un bruit constant additif. Ce test confirme une fois de plus que DIGIT ne peut pas être utilisé en relation avec les forces dans les expériences réelles sans une étude plus approfondie.

En outre, nous signalons d'autres problèmes que nous avons observés au cours des expériences avec le DIGIT, qui limitent fortement son utilisation pour la saisie répétée dans des expériences réelles : la couche d'élastomère est très fragile, elle s'use et se casse facilement, ce qui modifie les images de sortie ; les contacts avec le bord du capteur ne

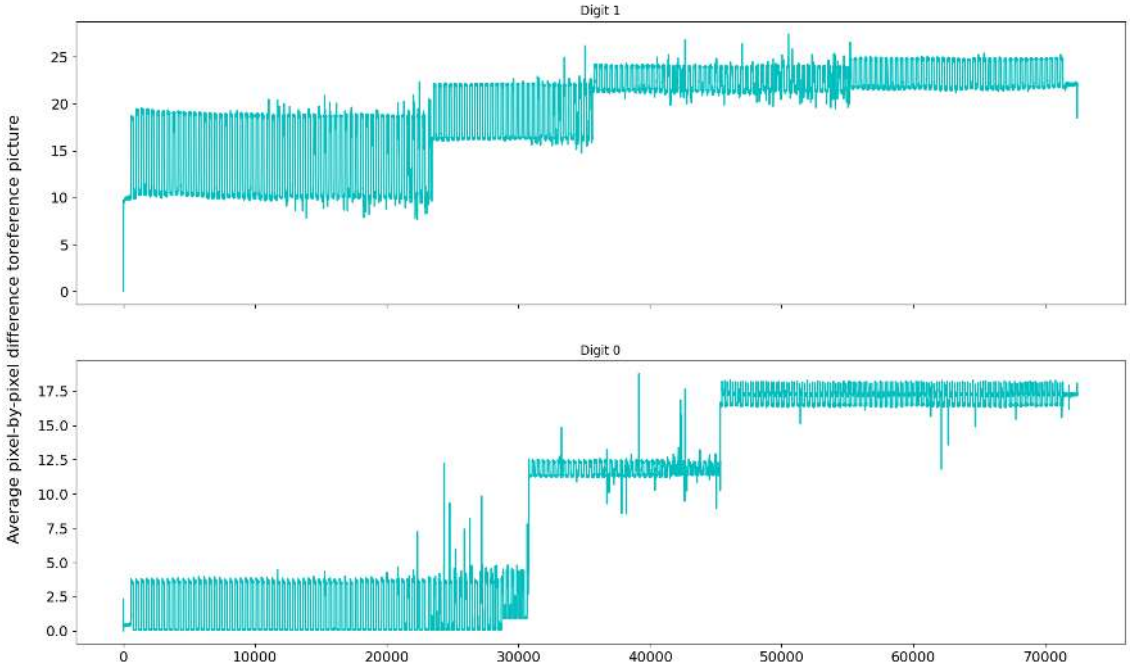


FIGURE 4.5 – Test 4 : sortie de DIGIT sous forme de δP en relation avec des contacts répétés de la même force, générés par la pince.

sont pas bien détectés ; enfin, les objets texturés ne sont souvent pas perçus par PyTouch, ce qui limite les performances de détection du glissement.

Pour toutes ces considérations, DIGIT ne semble pas être adapté aux expériences de préhension répétitives et de longue durée dans le monde réel. Il semble être indiqué pour une utilisation limitée à la détection du contact et à l'estimation de la surface de contact.

4.2 Procédure de validation de l'ellipsoïde de contact comme prédicteur de la réussite de la préhension

Suite aux observations de [23], nous émettons l'hypothèse que l'ellipsoïde de contact est un bon prédicteur du succès et de la stabilité de la saisie. Pour tester cette hypothèse, nous concevons la procédure suivante. Nous sélectionnons N objets, placés un par un sur l'espace de travail du robot au même endroit, et effectuons plusieurs saisies (par exemple, 7-10) avec différentes hauteurs de saisie, en appliquant la même force. Pour chaque saisie, nous évaluons deux paramètres : le succès de la saisie et la stabilité de la saisie. La saisie est réussie si l'objet est soulevé de la table et amené à une position fixe (0,55 cm au-dessus de la table).

La préhension est stable si l'objet ne glisse pas pendant ou après la procédure de stabilité GRASPA, proposée par [13] : l'objet est maintenu par la pince (sans autre serrage) pendant que l'effecteur se déplace pendant 40 secondes le long d'une trajectoire d'excitation constituée de roto-translations rapides, en particulier de rotations autour de l'axe de l'effecteur. Fig. 4.6 montre quelques postures pendant cette procédure. Pour chaque saisie,

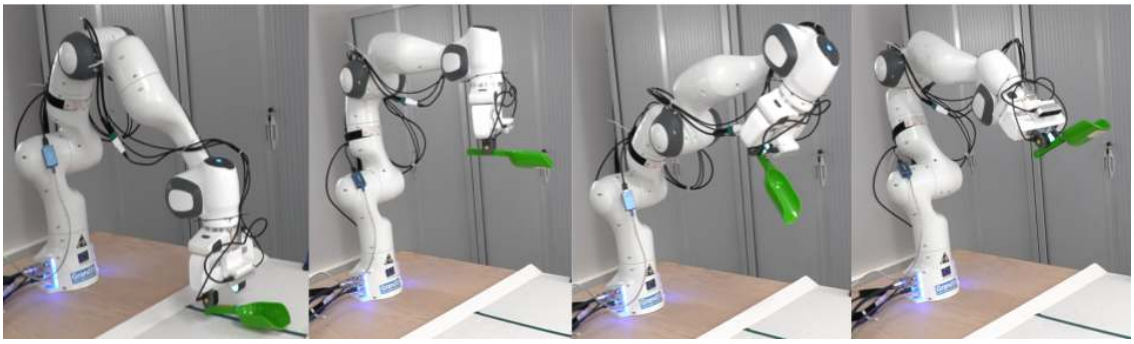


FIGURE 4.6 – Quelques images de la procédure de stabilité GRASPA. Une fois l'objet saisi, le robot le soulève et exécute une trajectoire d'excitation.

TABLE 4.1 – Matrice de confusion pour la *méthode de calcul de la surface de contact*, filtrant les ellipses trouvées pour les surface de contact sur la base des informations de préhension enregistrées pour chaque objet et expérience.

		Prediction label (ellipse presence)	
		Positive	Negative
True label (object grasped)	Positive	114	60
	Negative	7	15

nous enregistrons la sortie DIGIT, la surface de contact minimale ϵ mesurée par PyTouch *ContactArea*, et l'information binaire de réussite ou d'échec pour la réussite de la saisie et la stabilité. Notez que le calcul de l'ellipse de PyTouch échoue parfois et ne renvoie rien, par exemple, il y a un contact avec un objet mais il n'est pas détecté. Nous considérons alors les cas suivants : vrai positif : l'algorithme trouve une ellipse et la saisie est réussie ; vrai négatif : l'algorithme ne trouve pas d'ellipse et la saisie n'a pas réussi ; faux positif : l'algorithme trouve une ellipse et la saisie n'a pas réussi ; faux négatif : l'algorithme ne trouve pas d'ellipse et la saisie a réussi.

4.2.1 Expérience : stabilité de la préhension et ellipsoïde de contact

Dans cette expérience, nous voulons évaluer si l'aire de contact ϵ peut être utilisée pour prédire le succès et la stabilité de la saisie. Nous nous attendons à ce que des zones de contact ϵ plus élevées conduisent à des saisies plus stables. Nous avons sélectionné 10 objets avec différents niveaux de difficulté de saisie et évalué plusieurs saisies (jusqu'à 11, pour chaque objet) avec différentes hauteurs, en suivant la procédure décrite dans la section 4.2, pour un total de 192 images.

Le tableau 4.1 montre une valeur plus élevée pour les faux négatifs, ce qui signifie qu'il y a eu plus d'expériences où un objet a touché le capteur DIGIT, mais n'a pas été perçu par le traitement de l'image du capteur. En particulier, comme le montre le tableau 4.2, le boulon en plastique, qui a une forme particulière et une texture striée, a été "perçu" par le capteur (changement visible dans la sortie visuelle, du moins pour l'œil humain) mais PyTouch n'a jamais détecté d'ellipse. L'exactitude et la précision de la perception d'un objet étaient respectivement de 65,81% et 94,21%. Sur les 114 images réellement

TABLE 4.2 – Relation entre le taux de réussite de la saisie et du test de stabilité dans les tests physiques et le taux de réussite pour le calcul de l'ellipse par la méthode du *calcul de la surface de contact*, en considérant les données non filtrées.

object	Grasp success	Stability success	Ellipse calculated	
			Digit 0	Digit 1
bottle	9/11	5/11	9/11	9/11
dust shovel	7/9	6/9	7/9	7/9
screwdriver	7/8	7/8	7/8	6/8
white tube	6/8	5/8	7/8	7/8
green shovel (handle)	10/10	7/10	9/10	4/10
strawberry	7/7	7/7	3/7	3/7
green shovel (inside)	10/10	9/10	7/10	9/10
ear protector	6/9	6/9	5/9	1/9
golf ball	9/9	7/9	7/9	7/9
plastic bolt	8/8	8/8	0/8	0/8

positives, 14 présentaient des ellipses qui ne correspondaient pas au contour de l'objet. La Fig. 4.7 montre la relation entre l'aire de contact ϵ et la réussite et la stabilité de la saisie. Les résultats confirment qu'une surface de contact plus importante est associée à une probabilité plus élevée de stabilité et de réussite de la saisie.

4.3 Procédure de collecte de la meilleure hauteur de saisie

Les résultats de la section 4.2.1 montrent que la surface de contact de la prise ϵ est un bon prédicteur du succès de la prise, ce qui signifie que nous pouvons l'utiliser pour automatiser la recherche d'une bonne hauteur de prise.

Pour collecter une base de données sur la meilleure hauteur de préhension pour des candidats de préhension 3D donnés, nous concevons la procédure suivante. Nous collectons une série de saisies de haut en bas, où la position centrale de la pince (x, y) et l'orientation θ sont définies manuellement, et la profondeur initiale de la saisie (la position z de la pince) z_i est calculée en trouvant dans la hauteur du nuage de points la distance la plus proche de la caméra près de la position de saisie, comme cela est fait dans [56]. L'objet est ensuite saisi séquentiellement à 5 hauteurs différentes : 2cm au-dessus de z_i, 1cm au-dessus de z_i, à z_i, 1cm en dessous, 2cm en dessous. Les objets légers et instables sont maintenus en place, les hauteurs inférieures au plan de l'espace de travail sont ignorées.

Pour chaque prise, nous calculons la surface de contact ϵ de chaque DIGIT. La hauteur de la prise présentant les zones de contact combinées les plus élevées, z_c, est considérée comme la meilleure. La figure 4.8 illustre la procédure. La meilleure hauteur associée à chaque saisie 3D (x, y, θ), encodée par la représentation latente (VAE des saisies représentées par des patchs d'image) de [56], est ajoutée au jeu de données des démonstrations de saisie 4D, utilisé pour entraîner le régresseur de hauteur décrit à l'étape suivante. Nous

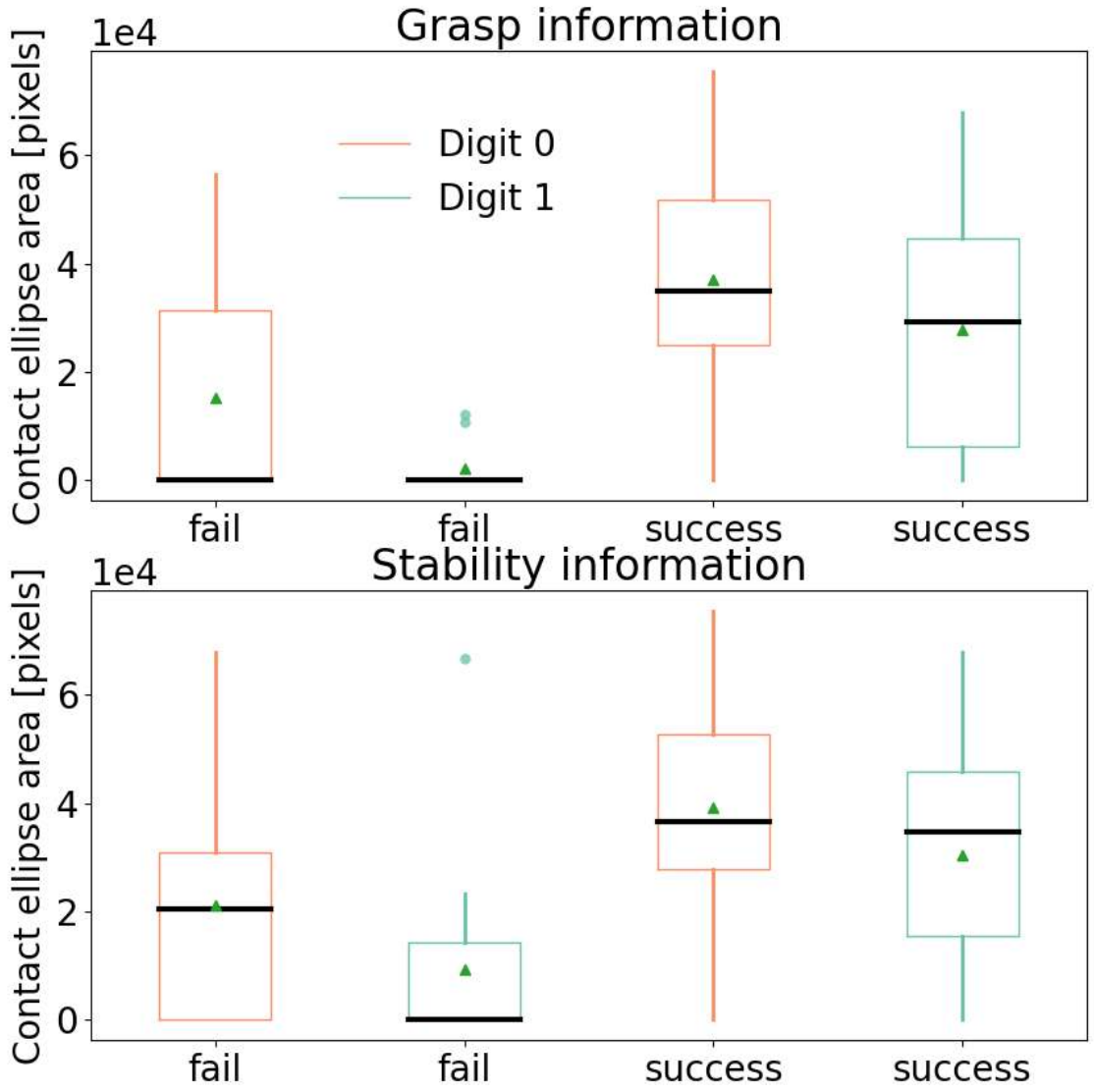


FIGURE 4.7 – Représentation graphique de la surface de l'ellipse de contact des deux DIGIT dans les cas d'échec et de réussite, pour les tests de saisie et de stabilité.

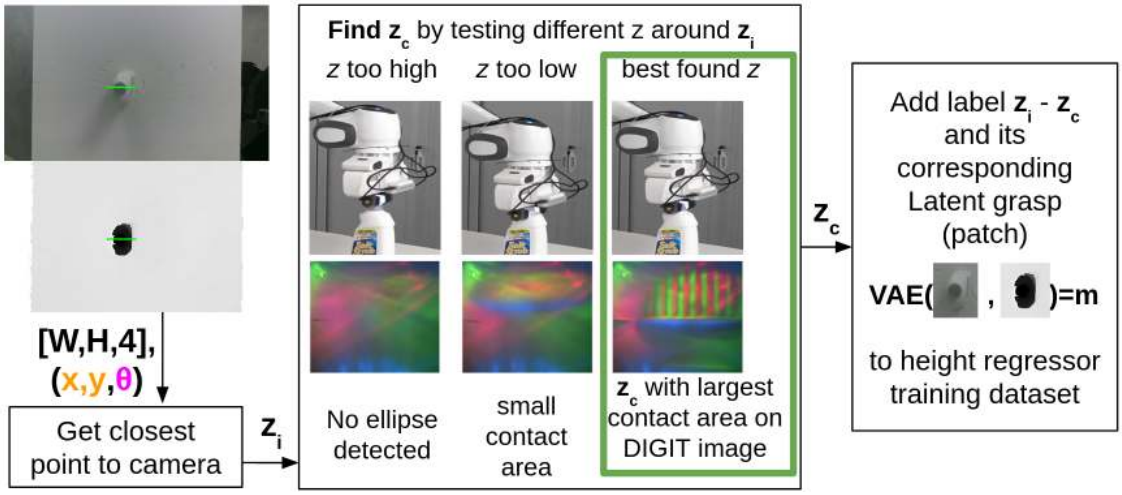


FIGURE 4.8 – Trouver la meilleure hauteur de saisie : 5 hauteurs sont essayées pour chaque saisie 3D, la hauteur associée au plus haut ϵ pour les deux DIGITs est retenue. La meilleure hauteur et sa saisie, encodée par le VAE de [56], est sauvegardée dans un jeu de données utilisé pour entraîner le régresseur de hauteur.

avons collecté 54 démonstrations de 10 objets de la base de données YCB (voir les 10 premières lignes du tableau 4.3) en utilisant cette collecte pilotée par capteur.

4.4 Entraînement de la prédiction de la hauteur de la prise

Pour une image RGB-D donnée $(W, H, 4)$, le jeu de données de la section précédente 4.3 contient des démonstrations de préhension, représentées dans les coordonnées de l'image par la position centrale du préhenseur (x, y), pivotées selon l'orientation θ et encodées réutilisant le même encodage de préhension de sous-section 3.3.2 du chapitre 3. Chaque saisie est représentée sous la forme d'un patch tourné $(w, h, 7)$ centré sur le milieu de la saisie, le patch est soumis à un Variational Auto-Econder (VAE) pour obtenir une représentation latente m. La représentation du patch est une représentation pratique héritée de [106], et l'encodage latent s'est révélé efficace en termes de données dans chapitre 3.

Cette représentation est ensuite utilisée pour entraîner un régresseur de hauteur qui apprend à prédire la correction c en mètres entre l'estimation initiale de la hauteur z_i et la hauteur z_c trouvée à l'aide des deux DIGIT (Sec. 4.3).

En principe, il n'y a pas de méthode préférable pour la conception du régresseur, nous avons donc comparé différentes méthodes : SVR [177], Random Forest [18], AdaBoost [51], processus gaussien [146], régression linéaire [109] et réseau neuronal [68]. L'entraînement peut être effectué hors ligne. En ligne, la hauteur initiale z_i est calculée à l'aide des données de profondeur de la caméra RGB-D : nous extrayons un patch orienté du nuage de points de profondeur avec la largeur de la prise sélectionnée et une hauteur fixe (5 pixels), et nous utilisons le point le plus proche de la prise (c'est-à-dire le point le plus haut de l'objet). On trouve z_c en ajoutant la sortie c du régresseur à z_i. Fig. 4.9 illustre comment le module de prédiction de la hauteur est incorporé dans le pipeline de préhension.

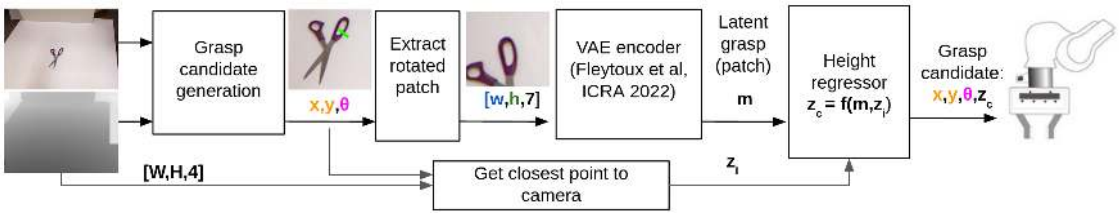


FIGURE 4.9 – Le pipeline de saisie, en supposant que le VAE et le régresseur de profondeur ont été entraînés auparavant et qu'un algorithme de génération descendante de candidats de saisie approprié est fourni (par exemple, GR-ConvNet [89], Dex-Net [108], ...). À partir d'une image RGB-D, un générateur de saisie produit un candidat de saisie. La hauteur initiale de la saisie z_i est calculée à l'aide de l'image de profondeur. Le candidat à la saisie est représenté par des patchs tournés centrés sur la position centrale de la pince (x, y). Il est envoyé à un VAE pour obtenir sa représentation latente m, qui est, à son tour, l'entrée du régresseur de hauteur (Sec. 4.4) entraîné sur le jeu de données collecté à l'aide de DIGIT (Sec. 4.3) pour obtenir la hauteur corrigée z_c.

4.4.1 Entraînement du prédicteur de la hauteur de la prise stable

En utilisant la procédure décrite dans la section 4.4, des prises stables sont utilisées pour entraîner un prédicteur de hauteur de prise. Nous avons effectué une validation croisée 4 fois pour comparer plusieurs modèles de régresseur (de Scikit-learn) avec différents paramètres trouvés à partir d'une recherche sur grille. Le même encodage d'entrée de sous-section 3.3.2 est employé, avec un VAE entraîné sur un jeu de données de 2349 scènes RGB-D (339 objets différents de celui utilisé dans l'expérience), avec des saisies générées à l'aide du GR-ConvNet [89], Dex-Net [108] et GPD [136].

Les modèles ont été entraînés avec une Nvidia GTX 1080 et un processeur Intel(R) Xeon(R) Gold 5118 à 2,30 GHz.

La Fig. 4.10 compare leurs performances : les résultats suggèrent qu'une régression linéaire simple est légèrement plus performante que les autres méthodes. Cette dernière se base sur une méthode qui cherche à minimiser la somme des carrés des erreurs entre les valeurs prédites (sur la ligne de régression) et les valeurs réelles (issues de l'ensemble de données). Il existe plusieurs raisons possibles à cette performance supérieure. Les modèles plus sophistiqués sont potentiellement plus susceptibles d'être sujets à une surotimisation ou une sous-optimisation. Ce phénomène peut résulter en une performance moindre si leurs hyperparamètres ne sont pas correctement ajustés. De plus, il se pourrait que les relations dans les données soient essentiellement linéaires, rendant la régression linéaire appropriée.

4.4.2 Test du prédicteur de la hauteur de la prise stable

Nous évaluons les performances du prédicteur de hauteur de préhension stable appris en saisissant les 10 objets YCB (indices 1 à 10) vus à l'entraînement, mais présentés dans de nouvelles positions, et 5 objets nouveaux et difficiles à saisir qui ne font pas partie de la base de données d'entraînement.

Pour évaluer l'impact du modèle de régresseur de la hauteur, nous comparons la réussite de la saisie et de la stabilité avec et sans la correction de la hauteur (c'est-à-dire

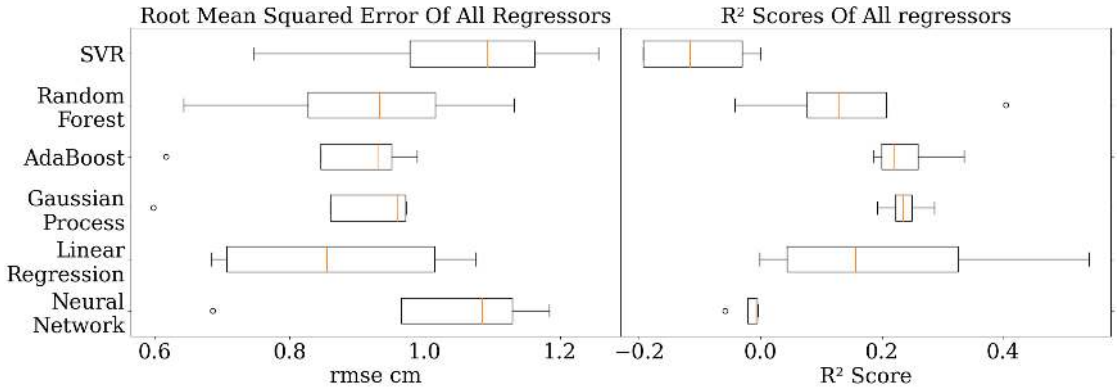


FIGURE 4.10 – Performances de différents modèles de régression en utilisant une validation croisée à 4 plis. Pour chaque pli, les modèles ont été entraînés avec les mêmes données d'apprentissage et de validation (en utilisant 75% de l'ensemble de données), une recherche en grille a été utilisée pour trouver des paramètres appropriés. Les résultats ci-dessus concernent les performances de chaque découpage sur leurs démonstrations de saisie de test restantes.

avec et sans le régresseur). Tous les objets ont été saisis à un emplacement (x, y) similaire dans l'espace de travail.

L'utilisation du régresseur a conduit à une amélioration de 5-6% sur 300 saisies (20 par objets) pour les objets vus et non vus, comme indiqué dans le tableau 4.3, ce qui montre que le régresseur est capable de généraliser à de nouvelles scènes d'objets vus précédemment et également à de nouveaux objets (indices 11 à 15). Bien que l'amélioration soit globalement très modeste, il convient de noter que la correction de la hauteur est cruciale pour permettre la saisie d'objets difficiles (indices 2-6-9-14-15), qui n'ont autrement que peu ou pas de chance d'être saisis avec succès, avec une amélioration de 16% pour la préhension et de 18% pour la stabilité.

4.5 Conclusions

Malgré ses limites, DIGIT peut être utilisé pour la collecte automatique de données sur les prises stables, car nous avons constaté expérimentalement qu'un ellipsoïde de contact plus élevé est associé à une meilleure réussite et stabilité de la prise. Nous avons utilisé DIGIT pour déterminer automatiquement la meilleure hauteur de saisie z pour les candidats à la saisie descendante (x, y, θ) qui, autrement, auraient recours à l'heuristique. L'amélioration globale en termes de succès et de stabilité de la saisie est relativement modeste pour tous les objets, ce qui signifie que l'heuristique est souvent suffisante pour la plupart des objets quotidiens. Cependant, notre prédiction de la hauteur devient significative pour les objets difficiles (avec des transparences, des formes courbes, etc.) qui seraient autrement très difficiles ou impossibles à saisir avec l'heuristique simple.

TABLE 4.3 – Résultats de l'expérience de saisie du robot avec et sans utilisation du régresseur de profondeur.

	with regressor		without	
object	Grasp success	Stability success	Grasp success	Stability success
1 YCB_screwdriver	10/10	10/10	10/10	10/10
2 YCB_power_drill	7/10	3/10	6/10	2/10
3 YCB_scissors	8/10	7/10	9/10	8/10
4 YCB_orange_plastic_bolt	10/10	10/10	10/10	10/10
5 YCB_adjustable_wrench	6/10	6/10	6/10	5/10
6 YCB_hammer	9/10	3/10	8/10	2/10
7 YCB_glass_cleaner	10/10	10/10	10/10	10/10
8 YCB_big_spring_clamps	10/10	10/10	10/10	10/10
9 YCB_bleach_cleanser	10/10	7/10	10/10	3/10
10 YCB_rope	10/10	10/10	10/10	10/10
11 ear protector	8/10	7/10	8/10	8/10
12 bottle	10/10	10/10	10/10	10/10
13 white tube	10/10	10/10	10/10	9/10
14 green_shovel (inside)	4/10	4/10	0/10	0/10
15 dust shovel	7/10	4/10	5/10	5/10

Chapitre 5

Évaluation des performances humaines en matière de saisie d'objets dans un tas

L'objectif de ce chapitre est d'analyser et de fournir une référence de la performance humaine pour la tâche de saisie d'objets dans un tas. Ces actions permettent d'établir une base de comparaison humaine, qui pourra ensuite servir de base pour évaluer et optimiser la performance des systèmes robotiques et automatisés dans le futur. Cette analyse est utile pour comprendre les limites et les capacités des machines et des systèmes de contrôle, et pour leur fournir des démonstrations et des instructions basées sur les performances humaines, afin de guider leur développement et d'optimiser leur efficacité dans la réalisation de tâches similaires. Les participants, incluant des experts humains et des opérateurs débutants, ont été chargés de fournir les préhensions nécessaires pour retirer tous les objets d'un bac (Fig. 5.2), le bras robotique Franka Emika Panda étant ensuite chargé d'exécuter la saisie. Les démonstrations ont été réalisées à l'aide de l'interface graphique décrite dans la section A.5, les participants ont fourni des préhensions 4DOF qui étaient ensuite exécutées par le robot (Fig. 5.3).

FIGURE 5.1 – Le tri d'objets inconnus est une tâche ayant de multiples applications potentielles qui répondent à d'énormes besoins sociétaux, comme le tri des déchets nucléaires ou d'autres matériaux dangereux[115]. Les objets sélectionnés pour l'expérience ont été choisis en raison de leur ressemblance avec les objets couramment rencontrés dans les déchets radioactifs contaminés issus de la maintenance ou de la mise à l'arrêt de sites nucléaires. Par rapport aux objets domestiques généraux, on trouve davantage d'objets industriels tels que des chaînes, des gants, des tuyaux et d'autres objets métalliques [172].

Les données collectées [8] proviennent de 21 utilisateurs différents (voir le Tableau 5.1) saisissant, avec le robot Franka Emika Panda, des objets d'un tas composé de 38 objets industriels différents (45 objets au total dans le tas en comptant les occurrences d'objets multiples Fig. 5.1). Cette base de données comprend 842 saisies réussies et 249 saisies échouées, ainsi que près de 20 heures de vidéos du robot effectuant les saisies. Chaque saisie est associée à une scène RGB-D et l'objet visé par l'utilisateur est également étiqueté (voir la section A.4 pour plus de détails).

8. `https://mybox.inria.fr/d/3b26abac62924fb589da/`

5.1 Protocole expérimental

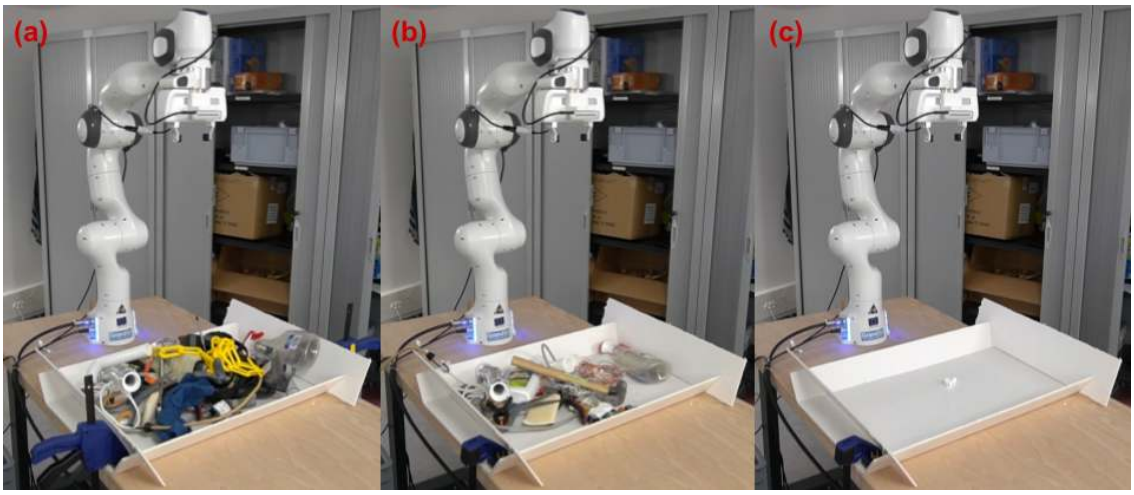


FIGURE 5.2 – Évolution du tas d'objets au cours d'une expérience : (a) au début, (b) au milieu et (c) à la fin.

Les participants avaient pour instruction de saisir un seul objet à la fois. Si l'objet visé n'était pas soulevé du bac, la saisie était considérée comme un échec. Si plusieurs objets étaient saisis en même temps, dont l'objet visé, la saisie était tout de même considérée comme un succès. En outre, certains objets (par exemple, le câble électrique) pouvaient nécessiter plusieurs saisies pour être retirés du bac en raison de leur taille ou de leur forme, ce qui entraînait plusieurs succès. Les participants n'étaient soumis à aucune contrainte de temps pour vider le bac.

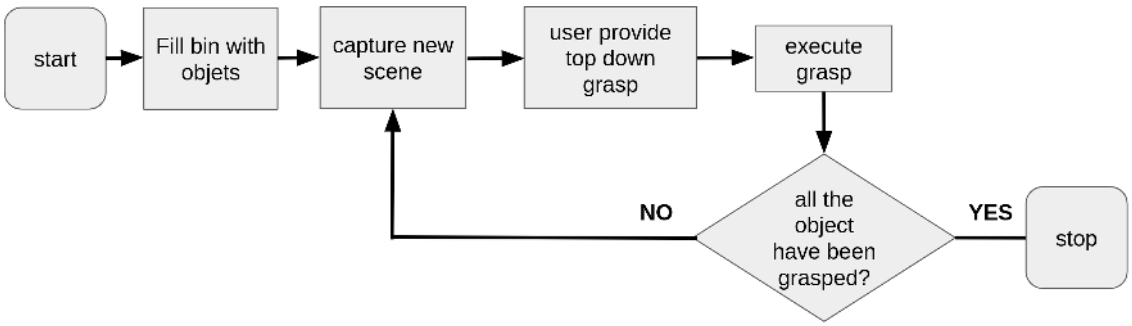


FIGURE 5.3 – Déroulement de l'expérience.

Le protocole expérimental a été conçu pour évaluer la performance des participants lors de la manipulation d'objets variés et complexes. Le déroulement de l'expérience, illustré dans la Figure 5.3, consistait à vider progressivement un bac contenant un tas d'objets, en effectuant des saisies successives jusqu'à ce que tous les objets soient retirés. Les participants utilisaient l'interface graphique décrite dans la section A.5 pour indiquer les préhensions 4DOF qui étaient ensuite exécutées par le robot. Cette procédure a permis

d'évaluer les stratégies adoptées par les participants, leur performances dans la saisie d'objets et les défis rencontrés lors de la manipulation d'objets dans un environnement encombré et complexe.

5.2 Analyse des résultats

La performance d'un système pour vider un bac est souvent évaluée en utilisant le taux d'achèvement. On considère qu'une série d'essais de vidage de bacs est réalisée, les bacs étant toujours chargés avec le même nombre d'objets et avec les mêmes objets. Le taux d'achèvement correspond à la proportion d'essais de vidage de bacs réussis, c'est-à-dire lorsque le robot a vidé l'ensemble des objets du bac. Dans notre cas, ce taux s'élève à 95% (un participant a dû interrompre l'expérience en raison de contraintes de temps, mais aurait probablement réussi à finir le bac avec plus de temps).

Les méthodes en SE(2), qui prédisent des saisies en 4 dimensions [120, 205, 157, 89, 210, 98, 79, 179], et celles en SE(3), qui prédisent des saisies en 6 dimensions [136, 99, 144, 173, 209, 190, 19, 21, 76, 8, 80], rapportent généralement des taux de réussite des saisies dans des tas (Clutter Pile Grasp success rates) compris entre 80% et 90%. Dans notre expérience, le taux de succès moyen est de 77,17%, ce qui est légèrement inférieur. Cette différence peut être attribuée à plusieurs facteurs : la complexité des objets sélectionnés, la configuration de l'expérience (voir Fig. 5.5) et la variabilité du taux de succès entre les participants.

L'analyse détaillée des résultats révèle des différences significatives entre les participants en termes de taux de succès et de temps nécessaire pour vider le bac. Ces variations peuvent être liées à l'expérience préalable des participants, à leur capacité à adapter leurs stratégies de saisie aux objets spécifiques et à la complexité de l'environnement. En comprenant mieux les facteurs qui influencent la performance des participants, nous pourrons identifier des pistes d'amélioration pour les algorithmes de préhension

5.3 Quels objets posent des difficultés ?

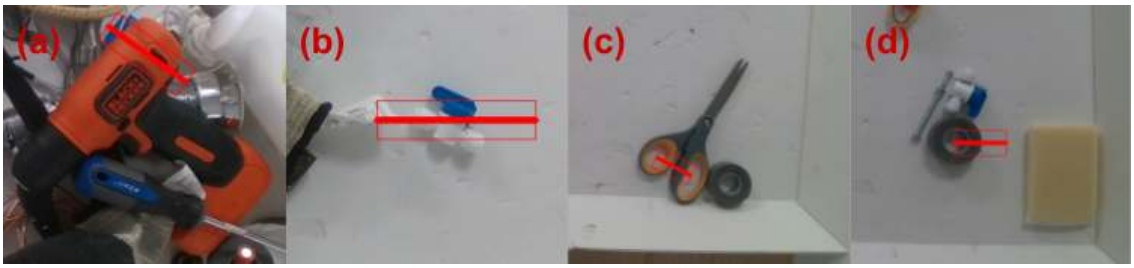


FIGURE 5.4 – Exemples de saisies ayant échoué.

Parmi les objets présentant les taux d'échec les plus élevés (voir les premières lignes du tableau 5.2), on distingue les outils (ciseaux, marteaux, perceuse électrique, pinces...) et les petits objets asymétriques (ruban adhésif, écrou, connecteur en plastique) (voir Fig. 5.4 a-b).

TABLE 5.1 – Récapitulatif des performances de chaque utilisateur (un parmi eux a dû arrêter l'expérience avant la fin pour des contraintes de temps), voir section 5.1.

user_name	positive_grasps	negative_grasps	time	success_rate (%)	finished
user1	29	8	0 :48 :30	78.378	✗
user2	40	19	1 :10 :29	67.796	✔
user3	37	12	0 :55 :45	75.510	✔
user4	39	11	0 :45 :23	78.0	✔
user5	43	10	0 :53 :14	81.132	✔
user6	38	9	0 :44 :40	80.851	✔
user7	42	16	1 :13 :15	72.413	✔
user8	40	20	0 :58 :45	66.666	✔
user9	39	21	1 :15 :41	65.0	✔
user10	41	10	1 :09 :35	80.392	✔
user11	40	3	0 :56 :07	93.023	✔
user12	42	6	0 :43 :26	87.5	✔
user13	40	18	1 :14 :40	68.965	✔
user14	40	9	0 :45 :54	81.632	✔
user15	39	13	0 :46 :32	75.0	✔
user16	40	19	1 :09 :06	67.796	✔
user17	44	16	0 :56 :36	73.333	✔
user18	44	1	0 :29 :56	97.777	✔
user19	41	10	0 :53 :57	80.392	✔
user20	41	6	0 :51 :01	87.234	✔
user21	43	12	1 :14 :48	78.181	✔

TABLE 5.2 – Récapitulatif des saisies exercées par objets, leur taux d'échec (le pourcentage de saisie sur ces objets qui ont échoué) et le nombre d'instance par objet dans le tas.

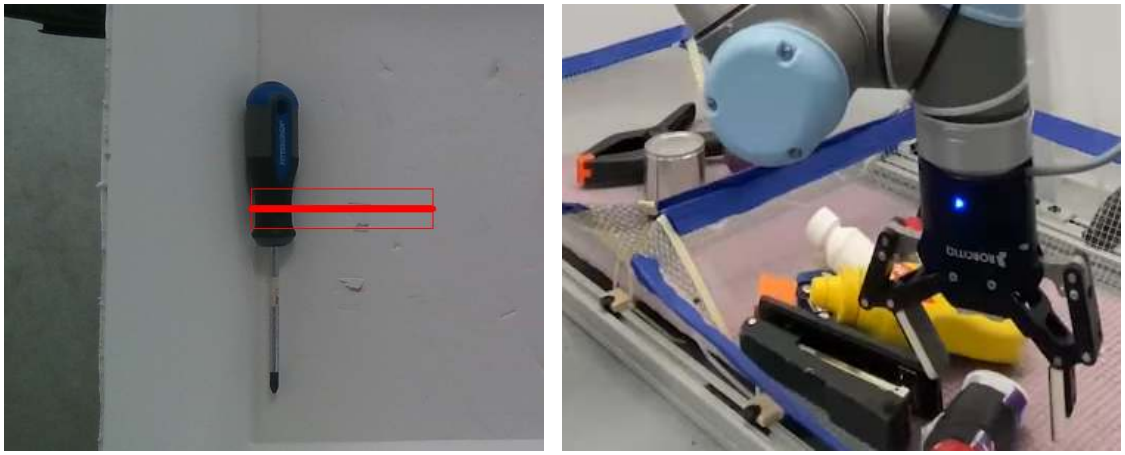
object_name	failure score (%)	grasps_success	grasps_failure	object_count
YCB_scissors	56,00%	20	28	1
YCB_hammer	42,11%	20	16	1
YCB_power_drill	41,67%	20	15	1
gray_tape	37,50%	23	15	1
YCB_bleach_cleanser	35,29%	20	12	1
YCB_adjustable_wrench	34,38%	20	11	1
YCB_orange_plastic_bolt	33,33%	34	17	2
red_plier	33,33%	18	9	1
connector	30,49%	57	25	3
YCB_big_spring_clamps	29,03%	22	9	1
YCB_grey_plastic_bolt	28,00%	18	7	1
YCB_tiny_spring_clamps	26,92%	19	7	1
black_glove	26,09%	17	6	1
metal_pipe	20,00%	22	6	1
electric_outlet	20,00%	20	5	1
YCB_screwdriver	19,23%	40	10	2
orange_tape	18,18%	34	8	2
YCB_rope	15,00%	17	3	1
security_glasses	14,29%	18	3	1
white_pipe	14,29%	42	7	2
electric_cable	13,33%	24	3	1
YCB_huge_spring_clamps	13,04%	20	3	1
black_earmuff	12,50%	19	3	1
bungee_cord	11,76%	15	2	1
transparent_glasses	11,11%	16	2	1
gardening_glove	10,53%	17	2	1
white_earphones	10,00%	7	1	1
battery	9,52%	19	2	1
YCB_plastic_bolts	9,52%	19	2	1
square_battery	9,09%	20	2	1
YCB_abrasive_sponge	9,09%	19	2	1
small_transparent_tube	8,33%	20	2	1
YCB_glass_cleaner	8,33%	21	2	1
bolt	5,26%	18	1	1
wires	2,56%	38	1	2
towels	0,00%	13	0	1
white_glove	0,00%	14	0	1
YCB_plastic_chain	0,00%	22	0	1

Les outils ont tendance à être plus lourds que les autres objets, en particulier par rapport aux objets dont les saisies n'ont jamais échoué (serviettes, gants, etc.).

Les petits objets asymétriques posent des difficultés en termes de précision en raison de leur taille et de leur forme. Les objets de tailles similaires, mais symétriques (boulon, pile) ou avec plus de tolérance vis-à-vis de la précision de la saisie (ruban adhésif orange) ne présentent pas ces difficultés.

Dans le cas des ciseaux et du ruban adhésif gris (voir Fig. 5.4 c-d), de nombreux échecs sont dus au fait que les participants n'ont pas pris en compte l'épaisseur des doigts du robot, rendant les saisies préféré par les utilisateurs difficiles car nécessitant une grande précision.

Une autre source de difficultés provient du fait que certains objets présentent un faible taux d'échec lors de la saisie, mais ont tendance à être saisis involontairement lors de la prise d'autres objets (écouteurs blancs, cordon élastique, gant blanc). Ceci est observable par leur nombre de saisies positives inférieur au nombre de participants (au total, on compte plus de 100 saisies ayant soulevé un objet supplémentaire).



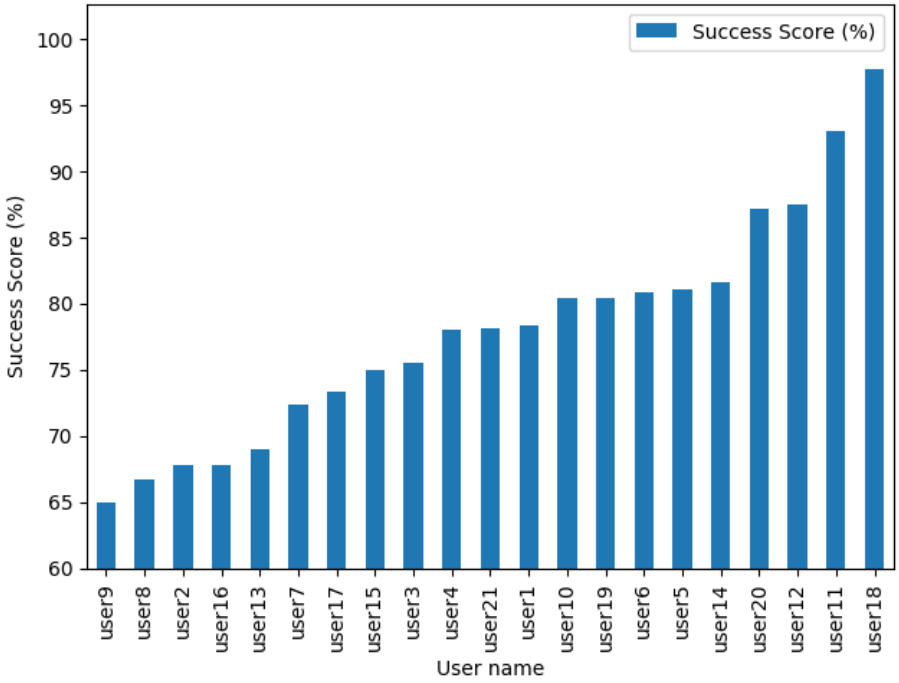
(a) Saisie proche du bord du bac ayant échoué. (b) Environnement de [210]

FIGURE 5.5 – (a) La saisie échoue en essayant d'éviter de rentrer en collision avec le bord du bac. (b) [210] évite ce genre de problèmes en utilisant la pince Robotiq 2F-85, dont les doigts se plient naturellement comme sur l'image, et un bac en filet non rigide.

Dans certains cas, les participants ont également rencontré des difficultés pour éviter que la caméra placée sur le préhenseur du robot ou les doigts du préhenseur n'entrent en collision avec les objets de l'environnement, provoquant un déplacement des objets à saisir et entraînant l'échec de la saisie. Les bords rigides du bac utilisés ont également causé des problèmes (voir Fig. 5.5).

On observe de grandes différences parmi les objets, avec des taux d'échec entre 56% et 0%.. Cette variabilité souligne l'importance de prendre en compte les caractéristiques spécifiques des objets, ainsi que les contraintes de l'environnement et du préhenseur, pour concevoir et comparer les algorithmes de saisie.

5.4 Performances des utilisateurs



(a) Histogramme des taux de succès par utilisateurs.

FIGURE 5.6 – (a) L'histogramme représente la distribution des taux de succès par utilisateurs, avec un gros groupe au centre de 12 utilisateurs dont les scores se situent entre 70% et 80%, un groupe de 5 utilisateurs en dessous de 70% et 4 utilisateurs au-dessus de 85%.

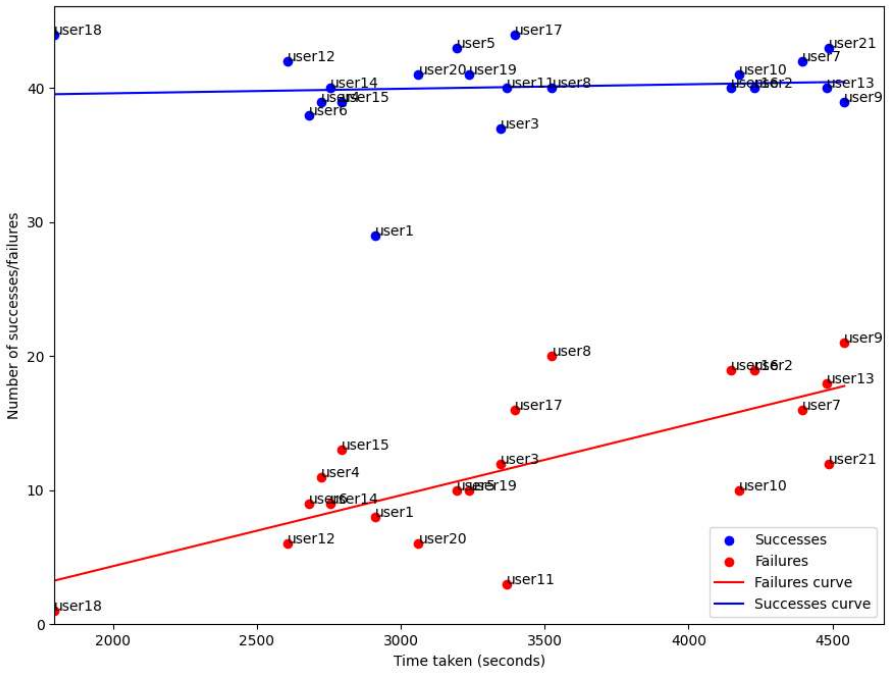
On observe également de grands écarts de performances entre les utilisateurs (voir Fig. 5.6 et Fig. 5.7) :

— Le temps passé sur l'expérience varie considérablement. L'utilisateur 18 a terminé l'expérience en seulement 29 minutes et 56 secondes, tandis que l'utilisateur 21 a pris plus de deux fois plus de temps, avec 1 heure, 14 minutes et 48 secondes.
— Les taux de réussite varient également considérablement d'un utilisateur à l'autre, allant de 65% pour l'utilisateur 9 à 97,77% pour l'utilisateur 18.

Définissons que les utilisateurs qui ont plus de succès sont ceux dont le nombre de succès par heure est supérieur à un écart-type (12,45) au-dessus de la moyenne (44,55) et que les utilisateurs qui ont moins de succès sont ceux dont le nombre d'échecs par heure est supérieur à un écart-type (4,53) au-dessus de la moyenne (12,06).

— Les utilisateurs ayant un taux de succès nettement supérieur sont ['user12', 'user18'].
— Les utilisateurs ayant un taux de succès nettement inférieur sont ['user8', 'user9', 'user15', 'user17']

De manière similaire, on peut considérer que les utilisateurs significativement plus rapides sont ceux dont le temps total (en secondes) pris est inférieur d'un écart-type (752) à la moyenne (3421), et les utilisateurs qui sont significativement plus lents sont



(a) Diagramme de dispersion des échecs/succès par rapport au temps.

FIGURE 5.7 – (b) Dans le nuage de points, nous observons une corrélation entre les deux variables, qui peut être interprétée comme une tendance des variables à évoluer ensemble. La ligne ajustée sur le nuage de points a clairement une pente, ce qui indique également la présence d'une relation entre les variables. De plus, nous avons également mesuré une corrélation de Pearson de 0,725. La corrélation de Pearson est une mesure de la relation linéaire entre deux variables, avec une valeur comprise entre -1 et 1. Une valeur de 0,72 suggère une forte corrélation positive entre les variables, ce qui signifie que lorsque le nombre d'échecs augmente, le temps pour réaliser la tâche tend également à augmenter.

ceux dont le temps pris est supérieur d'un écart-type à la moyenne.

— Les utilisateurs exécutant la tâche nettement plus rapidement sont ['user12', 'user18'].
— Les utilisateurs exécutant la tâche nettement plus lentement sont ['user2', 'user7', 'user9', 'user10', 'user13', 'user21']

Avec le nombre de saisies à l'heure (Picks Per Hour, abrégé PPH), le taux moyen est de 56,62 et l'écart-type à la moyenne (10,90) :

— Les utilisateurs choisissant plus rapidement les actions de saisie sont ['user18'].
— Les utilisateurs choisissant moins rapidement les actions de saisie sont ['user10', 'user21'].

La Figure 5.7 met en évidence une corrélation notable entre le nombre d'échecs de saisie et le temps requis pour terminer la tâche de vidage du bac. En examinant l'ensemble des résultats, il est intéressant de constater que si certains utilisateurs réussissent à respecter les consignes de l'expérience (peu d'échecs, réalisation rapide de la tâche, saisie d'un seul objet à la fois), d'autres utilisateurs rencontrent des difficultés sur certaines contraintes spécifiques.

Par exemple, sur la figure, lorsque le nombre de succès est moins élevé, il est plus probable que les utilisateurs aient saisi plusieurs objets en même temps. L'utilisateur 6 fait partie des plus rapides, mais n'a obtenu que 38 succès, contre une moyenne de 40,65. De même, l'utilisateur 11 a réalisé 40 saisies positives et seulement 3 saisies négatives, ce qui a abouti à un taux de réussite élevé de 93,023%.

Ces observations montrent que les performances des utilisateurs sont hétérogènes et que chacun peut rencontrer des difficultés spécifiques en fonction de sa manière d'aborder la tâche.

5.5 Stratégies de saisies

Examinons maintenant les séquences de saisie effectuées par les utilisateurs afin de tenter de comprendre les stratégies et le comportement adoptés par ces derniers.

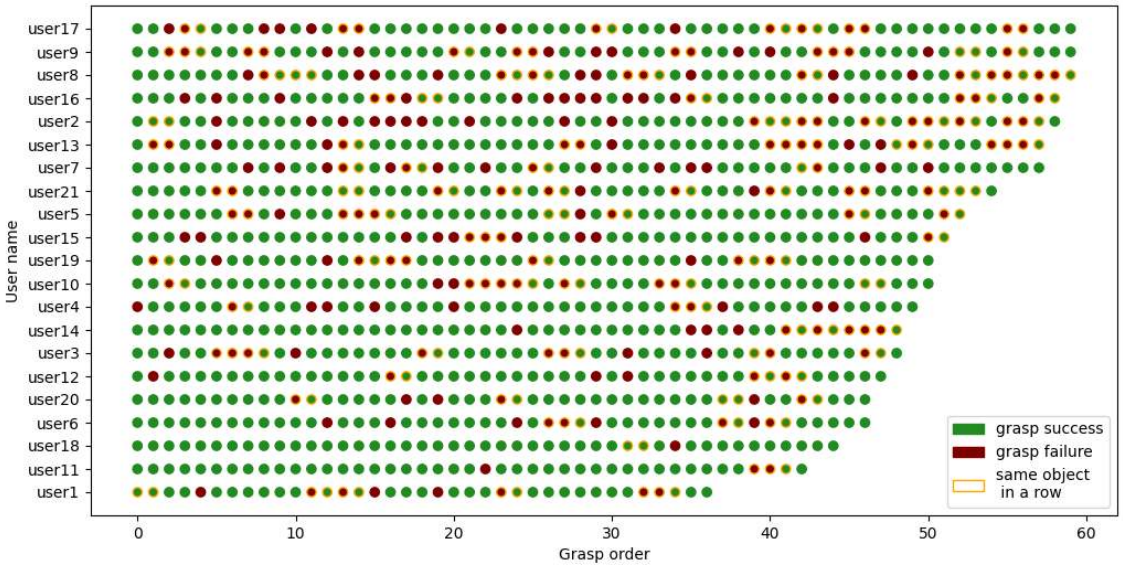


FIGURE 5.8 – Chaque point représente une saisie effectuée par un utilisateur. Le point est vert si la saisie est un succès, rouge si un échec, et son bord est orange si la saisie suivante ou précédente est sur le même objet.

La Figure 5.8 illustre la répartition des succès et des échecs parmi les saisies. Pour les expériences évaluant les performances des algorithmes de saisie sur tous les objets contenus dans un bac, il est courant de mesurer les taux de réussite de la saisie séparément pour les n premiers objets sur m. En effet, les algorithmes ont généralement pour objectif de saisir d'abord les objets les plus faciles à attraper, ce qui entraîne des taux de réussite de saisie plus élevés au début qu'à la fin. Dans notre étude, le taux d'échecs est également légèrement plus faible parmi le premier quart de saisie (un quart des saisies de chacun des utilisateurs) : 20,88%, alors que ce taux est de 27,31% pour le deuxième quart, 24,9% pour le troisième et 26,91% pour le dernier.

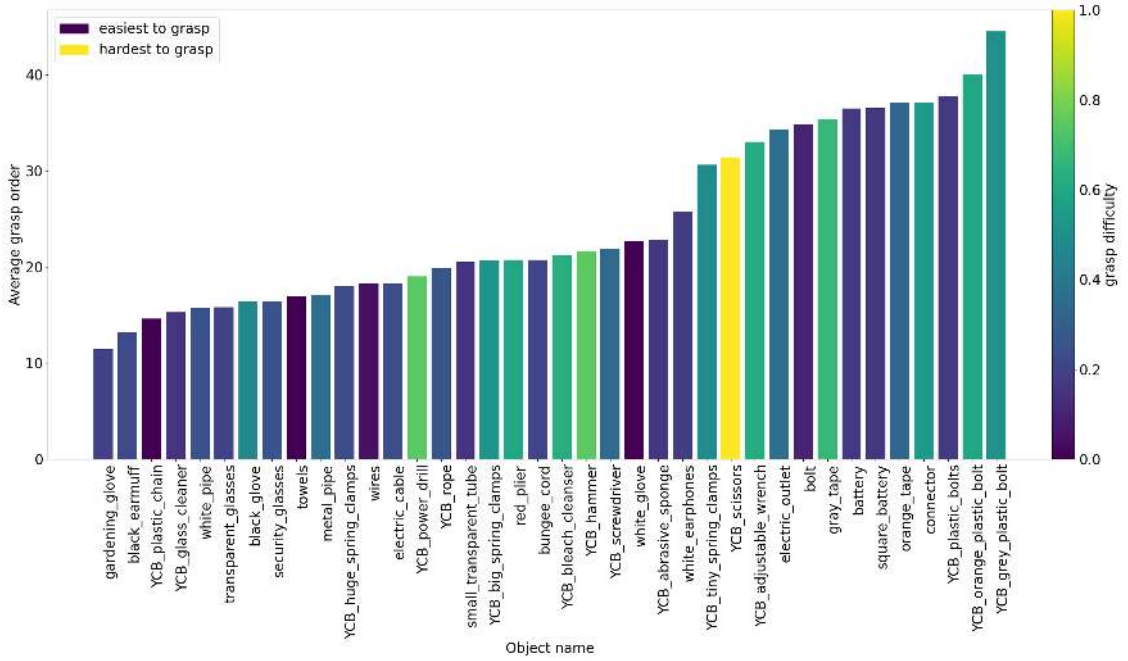


FIGURE 5.9 – L'axe des x représente chaque objet trié par ordre moyen de saisie, l'axe y correspond à cet ordre et chaque objet est associé à une teinte représentant sa difficulté (du bleu, les objets ayant causé le moins d'échecs de saisie, au jaune, les objets ayant causé le plus d'échecs).

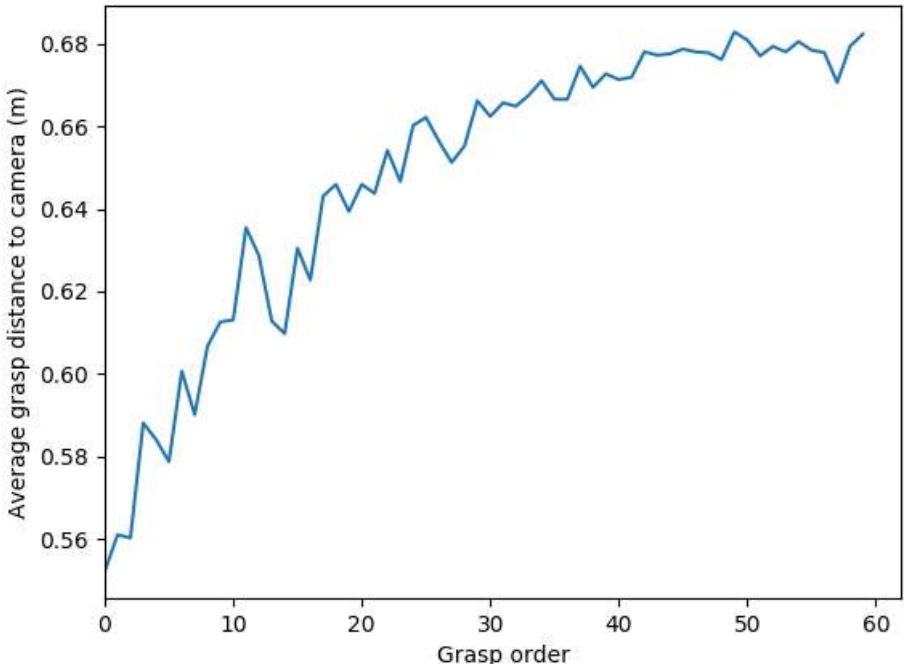


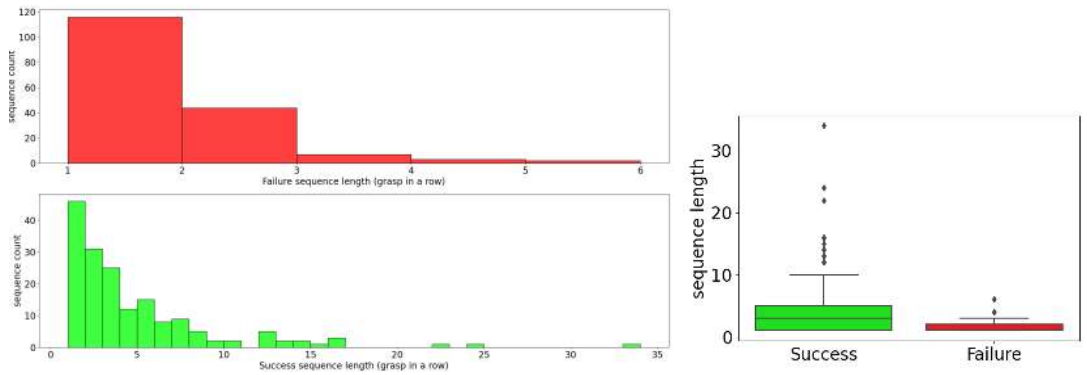
FIGURE 5.10 – Distance moyenne entre la caméra et l'objet en fonction de l'ordre de saisie.

Cependant, en classant les objets par ordre de saisie moyen par utilisateur (voir Figure 5.9), on constate que les objets les moins difficiles (ceux dont le taux d'échec est le plus bas, voir Tableau 5.2) sont effectivement saisis en premier : en moyenne, la difficulté des 50% premiers objets (taux d'échec de 27%) est 1,5 fois moins importante que celle de la seconde moitié (taux d'échec de 40,6%). Il est également à noter que les petits objets qui ont moins tendance à s'emmêler sont souvent saisis en dernier, car ils ont tendance à glisser au fond du bac. Cela se vérifie en affichant la distance moyenne en la caméra et l'objet en fonction de l'ordre de saisie : les objets les plus hauts du bac sont saisis en premier (voir Fig. 5.10).

En se focalisant sur les 9 utilisateurs ayant un taux de succès supérieur à 80%, on observe que le taux d'échecs du premier quart est de 12,5%, de 28,125% pour le second, de 26,5625% pour le troisième, et de 32,8125% pour le dernier. Cela montre clairement que la stratégie de saisir d'abord les objets les plus simples a été adoptée par les utilisateurs les plus performants.

Analysons si les individus persistent à essayer de saisir un objet jusqu'à réussir ou s'ils passent à un autre objet. Lorsqu'un utilisateur ne parvient pas à saisir un objet, il peut continuer à essayer de saisir le même objet jusqu'à réussir ou choisir de passer à un autre objet plus facile à manipuler. En cas d'échec, dans 55,665% (113) des cas, les utilisateurs passent à un autre objet et dans 35,96% (73) des cas, ils persistent jusqu'au succès. Dans 8,37% (17) des cas, ils poursuivent leurs efforts sur le même l'objet mais finissent par passer à un autre objet sans avoir saisi l'objet problématique.

Pour les 9 meilleurs utilisateurs, dans 46,15% (24) des cas, ils passent à un autre objet et dans 50% (26) des cas, ils persistent jusqu'au succès. Dans les 3,85% (2) des cas restants, ils retentent le même objet mais passent finalement à un autre objet sans avoir saisi l'objet problématique. **Cela suggère que la stratégie de persistance entraîne davantage de succès**.



(a) histogramme (b) boîte à moustaches

FIGURE 5.11 – Histogramme et boîte à moustaches des séquences successives d'échecs et de succès, ainsi que leur répartition. En moyenne, les utilisateurs réussissent à enchaîner 4 à 5 succès consécutifs et commettent rarement plus de 2 échecs d'affilée.

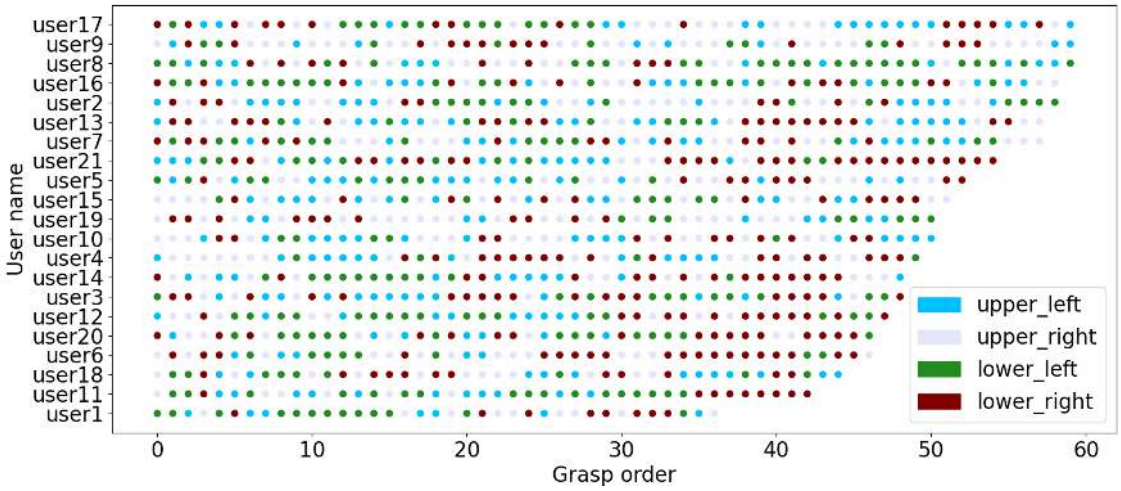
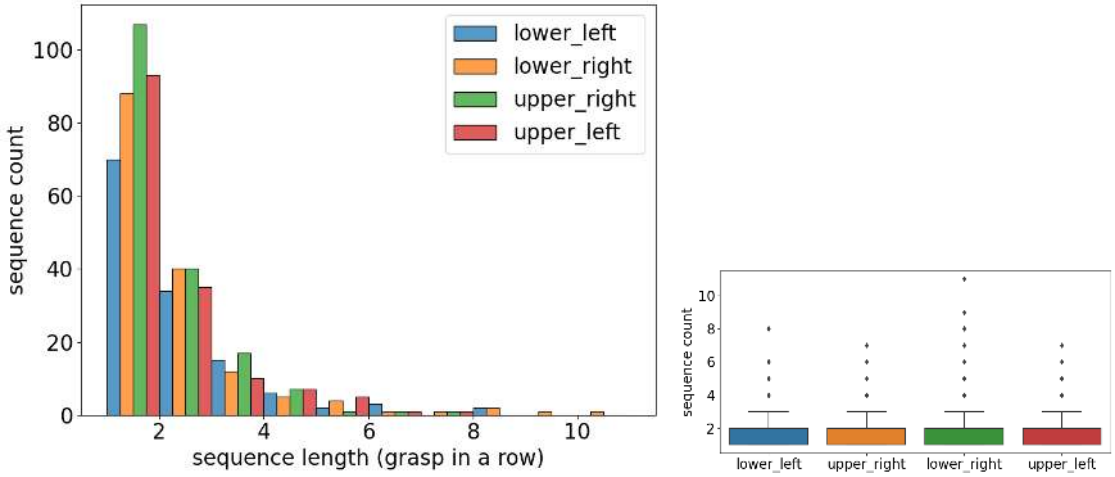


FIGURE 5.12 – Ordre de saisie en fonction de la position.

La figure 5.12 illustre les positions successives des saisies après avoir divisé le bac en quatre parties égales. En examinant la distribution des positions de ces saisies (Fig. 5.13), on constate que la moyenne des séquences de saisie par divisions est de 1,78 et de 1,82 pour les 9 meilleurs. **Cela suggère que les meilleurs utilisateurs ont plus tendance à choisir des saisies successives proche les unes des autres.** Cela peut indiquer que les utilisateurs les plus performants se concentrant sur des zones spécifiques du bac avant de passer à d'autres zones.



(a) histogramme (b) boîte à moustaches

FIGURE 5.13 – Histogramme et boîte à moustaches des séquences successives de saisie par division.

5.6 Conclusions

En conclusion, l'analyse de la performance humaine dans la tâche de saisie d'objets dans un tas offre des perspectives pour le développement et l'évaluation des systèmes robotiques de manipulation d'objets. L'étude des erreurs et des échecs humains aide à anticiper les défis auxquels les robots pourraient être confrontés et à optimiser leur fonctionnement en conséquence. Les performances des utilisateurs fournissent également une base de comparaison utile pour mesurer les performances des algorithmes.

L'analyse des résultats a mis en évidence l'impact du choix des objets sur la difficulté de la tâche, les grandes différences entre les utilisateurs et le fait que des saisies en 4DOF étaient suffisantes pour vider un tas complexe d'objets difficiles. Cependant, nous constatons que lorsque la majorité des objets a été évacuée du bac, les derniers objets subsistant se trouvent fréquemment à proximité immédiate des bords. L'utilisation de gestes non préhensibles pour les ramener vers le centre du bac pourrait considérablement faciliter les dernières saisies. L'analyse des stratégies des meilleurs utilisateurs pourrait également profiter aux algorithmes apprenant à partir d'exemples obtenues par des utilisateurs, en proposant des pistes de bonnes stratégies à suivre.

Chapitre 6

Comparaison des algorithmes de préhension et apprentissage des préférences de saisies de façon interactive

Dans ce chapitre, nous cherchons à proposer une version interactive de LGPS (voir chapitre 3) en association avec des générateurs de préhension récents. Grâce à cette approche interactive, le robot peut s'adapter aux besoins spécifiques de chaque utilisateur ou tâche, en tirant parti de la connaissance des générateurs de préhension (futurs ou existants), nécessitant peu de nouvelles données d'apprentissage.

Nous débutons ce processus par une comparaison détaillée des différents générateurs de préhension existants. L'objectif est d'identifier l'algorithme le plus performant, qui servira de point de départ pour le développement de la configuration "human-in-the-loop". La comparaison des algorithmes de préhension fournit une base solide à partir de laquelle nous pouvons améliorer les performances du robot grâce à l'apprentissage interactif. Ces deux éléments sont liés et contribuent ensemble à l'amélioration de la performance en matière de préhension. Cela permet également de documenter les performances de ces algorithmes avant l'intégration de l'apprentissage interactif, afin de quantifier l'amélioration apportée par celui-ci. Nous examinons en particulier les aspects suivants :

— Rapidité : Combien de temps faut-il pour générer et traiter les candidats à la préhension ?
— Taux de couverture des préhensions : Le taux de couverture est une métrique utilisée pour évaluer la performance des algorithmes de préhension. Il représente la proportion de toutes les préhensions possibles d'un objet détectées par l'algorithme. Cette métrique est similaire au rappel dans un classificateur (quelle proportion de vrais positifs est correctement identifiée) et est étroitement liée au taux de réussite des préhensions. Cependant, la nature continue de l'espace des préhensions possibles pour un objet rend difficile la comparaison des algorithmes en utilisant cette métrique, sauf si l'ensemble des préhensions possibles est défini pour chaque objet (comme dans le cas du jeu de données Cornell).
— Qualité de la préhension : Quelle est la probabilité de réussite des préhensions

générées par l'algorithme (l'objet est soulevé sans tomber et sans glisser hors de la prise, ou la préhension satisfait la métrique du rectangle) ?
— Généralisation : Comment se comportent les algorithmes face à de nouveaux objets et angles de caméra qu'ils n'ont pas encore rencontrés ?

Une fois le générateur de préhension le plus approprié identifié, nous abordons quelques adaptations nécessaires pour rendre l'algorithme utilisable de manière interactive. Nous examinerons également si l'apprentissage actif peut améliorer l'efficacité de l'entraînement du modèle, en diminuant le besoin de données étiquetées manuellement, tout en améliorant la qualité des prédictions.

6.1 Présentation des algorithmes de génération de préhension

En tant que partie prenante du projet HEAP, l'équipe de l'IIT a développé un cadre logiciel en Python utilisant ROS pour exécuter différents algorithmes de préhension sur le bras du panda. Le cadre permet de collecter les données de la caméra, les demandes de l'opérateur pour initier une préhension, et enfin, d'envoyer des commandes de haut niveau pour contrôler le robot. Le "Grasping Benchmarks Manager" gère toutes ces informations et envoie finalement les poses de saisie au serveur de service de contrôle du panda pour contrôler le mouvement du robot par ROS et Moveit. La structure du cadre suit celle présentée dans la Figure 6.1.

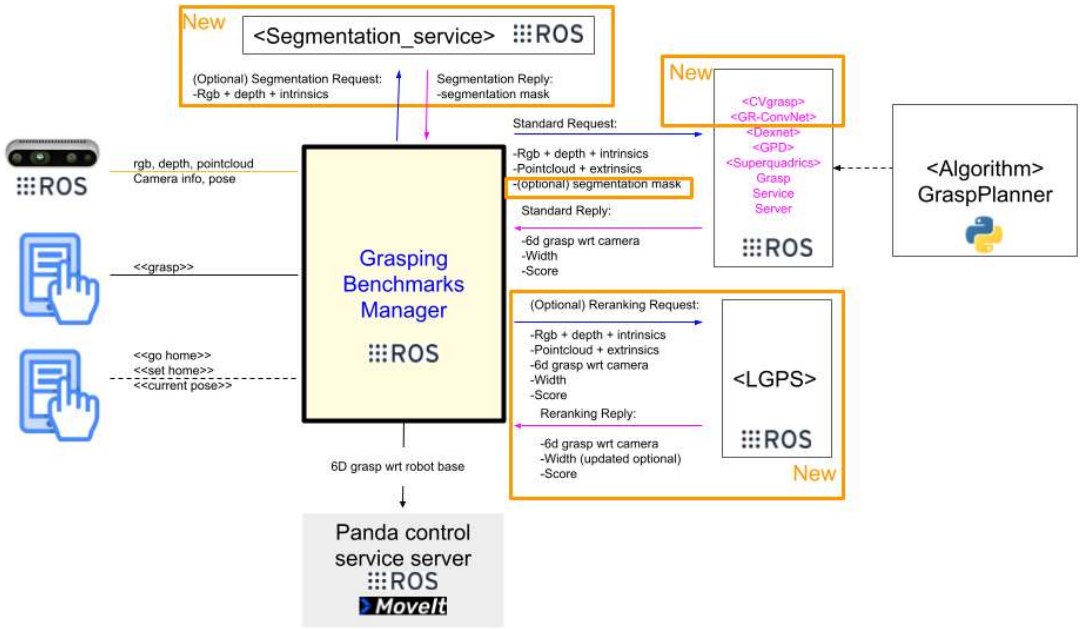


FIGURE 6.1 – Les serveur de générations de préhensions basé sur ROS utilisent une interface commune Python pour offrir les services de planification de saisie à d'autres nœuds par le biais d'une requête standard (contenant des données visuelles et les paramètres de la caméra) et une réponse (candidats à la saisie 6D).

L'INRIA et l'IIT ont collaboré pour intégrer deux générateurs de préhension supplémentaires (GR-Convnet (Sec. 6.1.3), CVGrasp (Sec. 6.1.4)) et l'algorithme LGPS [56]. L'algorithme LGPS est utilisé pour reclasser les candidats à la préhension fournis par les générateurs de préhension et sera présenté dans le chapitre suivant. Un service de segmentation 2D a également été implémenté et intégré pour découper l'objet cible du plan de la table. Cette fonctionnalité est conçue pour améliorer la qualité des poses de préhension générées par les algorithmes. La figure 6.1 présente une vue d'ensemble de l'architecture logicielle, avec ses principaux modules complémentaires. Cette architecture permet de générer des candidats à la préhension avec des combinaisons d'algorithmes supplémentaires : **1-** GR-Convnet et LGPS, **2-** CV et LGPS, **3-** Dex-Net (Sec. 6.1.1) et LGPS, et **4-** GPD (Sec. 6.1.2) en utilisant une configuration GPD appropriée pour filtrer la direction d'approche des candidats à la préhension et LGPS.

Chaque algorithme est utilisable à travers des images Docker, un outil qui permet de déployer facilement une application dans un format conteneurisé (les applications ne partagent que le noyau du système d'exploitation hôte). Tous les composants et dépendances nécessaires à une application sont regroupés dans un seul conteneur, pouvant être exécuté sur toute machine sur laquelle Docker est installé. Cette méthode facilite le développement, le test et le déploiement d'applications, car cela assure que l'application fonctionnera de la même manière sur n'importe quelle machine sur laquelle elle est déployée. Cette méthode d'installation est particulièrement adaptée dans notre cas, car les algorithmes et les paquets ROS nécessitent de nombreuses dépendances, dont certaines doivent être compilées manuellement à partir des sources et nécessitent des fichiers de configuration spécifiques, certains algorithmes utilisant des bibliothèques incompatibles entre elles.

L'empaquetage et le déploiement par le biais d'images de conteneurs résolvent ce problème en isolant chaque algorithme et ses dépendances dans un conteneur distinct. Ainsi, il est possible de déployer plusieurs algorithmes avec des dépendances incompatibles sur la même machine, tout en garantissant que chaque algorithme fonctionnera correctement dans son propre conteneur. Cette approche facilite également la mise à l'échelle de la solution en permettant de déployer plusieurs instances de chaque algorithme en fonction de la charge de travail. En somme, l'architecture ROS des algorithmes de génération de préhension déployée autour du "Grasping Benchmarks Manager" permet d'intégrer facilement de nouveaux algorithmes et de les rendre compatibles avec l'interface graphique, offrant ainsi une grande flexibilité dans le choix des algorithmes de préhension à utiliser en fonction des besoins de l'utilisateur.

6.1.1 Dex-Net

Le Grasp Quality CNN (GQ-CNN) ou Dex-Net 2.0 [106] est un détecteur de préhension qui nécessite une configuration dans laquelle la caméra est montée au-dessus de la scène (50-70 cm) avec son plan d'image parallèle au plan arrière de la scène (par exemple, la table ou le sol du bac), et produit des préhensions planes à 4 DoF (c'est-à-dire que l'axe d'approche est orthogonal au plan horizontal de la scène) qui sont paramétrées avec une position 3D et un angle autour de l'axe d'approche.

Il s'agit essentiellement d'un pipeline de préhension basé sur l'échantillonnage (Sec. 2.5.2.2) : la sélection de la préhension est basée sur l'élagage, la notation et le classement

d'un ensemble de préhensions antipodales.

L'échantillonnage des candidats est réalisé via la Cross Entropy Method (CEM) : des cycles itératifs d'échantillonnage et d'évaluation des candidats de préhension sont réalisés en utilisant le modèle. Après chaque tour d'évaluation, les meilleures préhensions sont sélectionnées, puis un nouvel échantillonnage est effectué à proximité des échantillons retenus.

Le CNN qui attribue les scores est entraîné sur une base de données de préhension synthétique (Sec. 2.5.1.2) représentée par des images de profondeur. La sortie du modèle est une prédiction binaire de la qualité de la préhension et est entraînée en utilisant une perte d'entropie croisée (cross entropy loss).

Dex-Net a connu différentes révisions et l'ajout de fonctionnalités au cours des dernières années, notamment l'évaluation de la préhension d'effecteur à ventouse (Dex-Net 3.0) [107] et la configuration bimanuelle avec des préhenseurs à mâchoires parallèles et à ventouse (DexNet 4.0) [113].

6.1.2 GPD

Grasp Pose Detection (GPD) proposée par Pas et al. [136] détecte les préhensions à 6 DoF sur des nuages de points, en supposant une pince à mâchoires parallèles.

Elle fonctionne en échantillonnant uniformément le nuage de points (qui peut être partiel ou complet) et en créant un candidat à chaque emplacement échantillonné en fixant la direction d'approche à la normale à la surface. Les candidats à la préhension sont filtrés à l'aide d'une liste de critères géométriques tels que la direction d'approche, la taille de la pince, l'ouverture maximale et l'espace de travail souhaité.

Afin d'échantillonner une préhension, cette approche échantillonne d'abord un point du nuage, puis estime le repère de Darboux (défini par les courbures principales d'une surface) à l'échantillon. Un candidat à la préhension est ensuite échantillonné à l'aide d'une heuristique définie par rapport au repère de Darboux. Les candidats pour lesquels la pince entre en collision avec des points du nuage ou qui ne contiennent pas de point du nuage dans la zone de fermeture de la pince sont élagués.

Finalement, la qualité de chaque candidat est estimée en considérant les points du nuage qui tombent dans le volume saisi par la pince et trois projections orthographiques pour obtenir un tenseur qui est introduit dans un petit modèle convolutif. La sortie du CNN est la qualité de la préhension, et il est entraîné sur un certain nombre de préhensions synthétiques.

6.1.3 GR-Convnet

Morrison et al. [120] ont développé GGCNN, un modèle de réseau neuronal convolutif (CNN) qui prend en entrée une image de profondeur et génère en sortie trois images représentant la qualité de la saisie, la largeur de la pince et l'orientation de la saisie pour une préhension au niveau de ce pixel. Entraînée sur la base de données de Cornell [75], cette architecture permet d'effectuer des préhensions robotiques de bout en bout en boucle fermée à une fréquence de 50 Hertz (Sec. 2.1.6).

Dans la même lignée que le GG-CNN, mais avec des performances supérieures, Kumra, Joshi et Sahin (2020) [89] ont conçu un modèle résiduel profond et plus large appelé GR-Convnet. Ce modèle est entraîné sur la base de données de Cornell [75] ou de Jacquard

[49], en utilisant des données RGB et/ou de profondeur (D). Tout comme le GG-CNN, cette architecture permet d'effectuer des préhensions robotiques de bout en bout en boucle fermée à une fréquence de 50 Hertz.

6.1.4 Approche basée sur la vision par ordinateur (CVGrasp)

L'algorithme LGPS mentionné précédemment (présenté au chapitre 3) classe les préhensions fournies par les algorithmes de préhension. Au cours de ces expériences, il a été constaté que Dexnet échoue sur 20 à 25% des scènes pour générer des candidats sur les scènes de la base de données de Cornell (section 6.2). Pour pallier ce problème, nous avons développé un algorithme simple de synthèse de préhensions basé sur la vision par ordinateur. L'algorithme prend en entrée un masque de segmentation spécifique à une image RGB-D de l'objet à saisir (Sec. A.6) et peut générer des préhensions parmi :

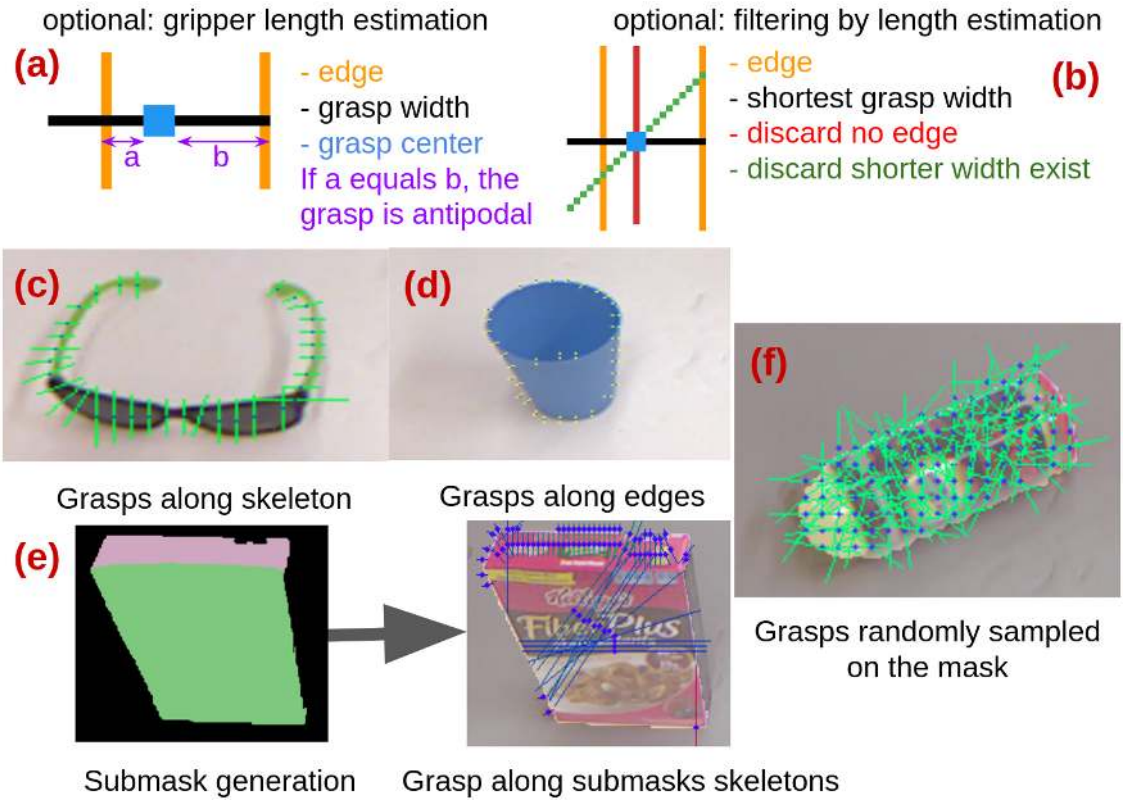
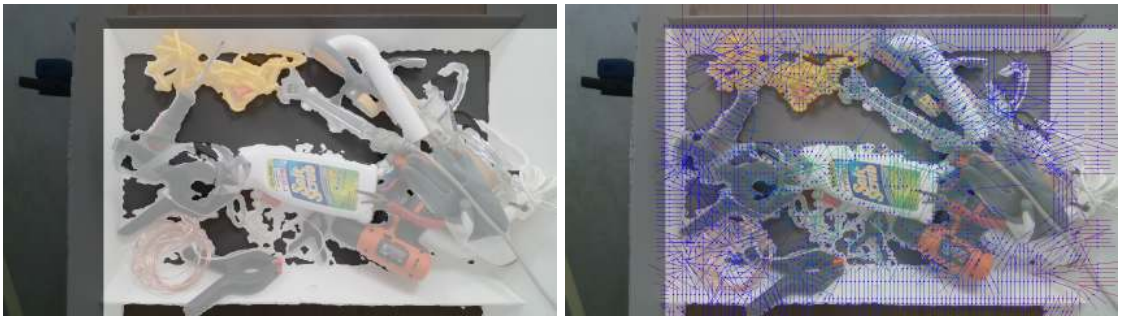


FIGURE 6.2 – Exemples de préhensions générées par CVGrasp.

— Les bords du masque après les avoir détectés en utilisant un détecteur d'arêtes Canny (Fig. 6.2-(d)) [30].
— Le squelette [94] de l'objet similairement à [181] (Fig. 6.2-(c)).
— Ou uniformément sur le masque (Fig. 6.2-(f)). Pour l'angle, 3 configurations sont possibles : 1 - choix aléatoire, 2 - test de chaque angle espacé d'un pas fixe et conservation de l'angle qui aboutit à la plus faible ouverture de la pince, 3 - conservation de chaque angle espacé d'un pas fixe.

La profondeur de la prise (la position z de la pince) est calculée en utilisant l'image de profondeur. L'orientation de la préhension et l'ouverture du gripper peuvent être trouvées via le masque, sinon des orientations et des ouvertures par défaut sont utilisées (Fig. 6.2-(a-b)).

Si nécessaire, des sous-masques peuvent être extraits en fonction des couleurs de l'image RGB ou de la géométrie de la profondeur (Fig. 6.2-(e)) [9], ce qui permet de générer des préhensions à des endroits spécifiques en exécutant la génération sur ces masques (par exemple pour le sommet de la boîte de céréales Fig. 6.2). La génération étant directement liée à la segmentation, la qualité de cette dernière affecte la génération (Fig. 6.3), en particulier lorsque plusieurs objets sont présents dans la scène.



(a) masque (b) candidats

FIGURE 6.3 – Dans ce tas dense, la segmentation retient beaucoup de points inutiles, la génération sur ce genre de scène, bien que fonctionnelle (2197 préhensions générées), excède les 10 secondes contre moins de 0.3 seconde pour une scène ne comportant qu'un seul objet.

Afin d'être compatible avec l'architecture du "Grasping Benchmarks Manager", un score de qualité peut également être attribué à chaque préhension en prenant en compte sa distance au centre du masque, sa largeur d'ouverture du préhenseur et sa distance à la caméra. Les paramètres à fixer pour la génération sont donc :

— Les poids de chaque paramètre de la fonction de qualité (optionnel).
— Quelle type de préhension à générer : squelette et/ou bord et/ou uniforme.
— L'espace en pixels entre chaque préhension et le pas entre chaque angle si le type uniforme est choisi.

En conclusion, l'approche basée sur la vision par ordinateur (CVGrasp) offre une solution alternative pour générer des préhensions lorsque d'autres méthodes, telles que Dexnet, rencontrent des difficultés. Bien que cette méthode présente des limitations, notamment en ce qui concerne la qualité de la segmentation, elle demeure une option intéressante pour les scénarios où les approches basées sur l'apprentissage profond ne sont pas applicables.

9. `https://scikit-learn.org/stable/auto_examples/cluster/plot_color_quantization.html`

6.2 Comparaisons sur la base de données Cornell

Nous testons, avec notre implémentation et sur notre matériel (processeur Intel(R) Core(TM) i7-8850H CPU @ 2.60GHz - carte graphique Quadro P2000 CUDA 11.5), chaque algorithme de génération sur la totalité du Cornell dataset (885 scènes), en utilisant des paramètres aussi proches que possible des paramètres par défaut. Les modifications apportées sont les suivantes :

- Dex-Net [106] : nous testons deux versions, l'une avec les paramètres par défaut, et l'autre avec des paramètres modifiés pour échantillonner davantage de candidats, encourageant ainsi une plus grande diversité.
- GR-ConvNet [89] : ce modèle prend en entrée des images de 224x224. Dans leurs expériences, les auteurs rognent l'image sur le centre x,y des étiquettes positives pour chaque scène. Dans nos expériences, comme nous n'avons pas accès aux étiquettes, l'image est rognée au centre du masque de segmentation (voir section A.6). GR-ConvNet propose trois modèles : un modèle entraîné sur les données RGB-D de Cornell, un modèle entraîné sur les données RGB-D de la base de données Jacquard et un modèle entraîné sur les données de profondeur de la base de données Jacquard. Étant donné que nous réalisons des tests sur la base de données de Cornell, nous ne testons que les modèles entraînés sur Jacquard.
- GPD [136] : ce modèle peut produire des préhensions en 6DOF. Les candidats sont filtrés pour ne conserver que les préhensions quasi perpendiculaires à la caméra. Les dimensions de l'espace de travail de GPD sont modifiées pour être adaptées aux scènes de Cornell, où les objets se situent à plus de 1,50 m de la caméra.

En effectuant ces tests sur la base de données de Cornell, nous pouvons comparer les performances de chaque algorithme de génération dans des conditions similaires, en tenant compte des spécificités de chaque modèle et des ajustements nécessaires pour s'adapter aux scènes de Cornell. Cela permettra d'évaluer l'efficacité de chaque algorithme et d'identifier les meilleures approches pour la génération de préhensions dans ce contexte.

Pour chaque générateur, nous comparons quatre scénarios.

Table 6.1 – Résultats des différents tests sur la base de données de Cornell.

generator name	Dex-Net	Dex-Net tweaked	GR-ConvNet jacquard d	GR-ConvNet jacquard RGB-D	GPD	CVGrasp
planification (s)	2.5	12	0,12	0,18	1,4	0,27
grasp generated	47018	432608	1754	2554	346349	612314
average grasp generated per scene	53	489	2	3	391	692
coverage rate	1264/5110	2530/5110	576/5110	598/5110	3280/5110	**4775/5110**
rectangle metric	366/885	389/885	221/885	237/885	358/885	**704/885**
max rectangle metric	505/885	707/885	251/885	262/885	806/885	**884/885**
with LGPS	138/244	183/244	96/244	87/244	215/244	**233/244**

Test 1 : taux de couverture.

Pour chacune des 885 scènes de la base de données de Cornell, et pour chaque saisie positive étiquetée (5 110 au total), nous testons si au moins une saisie générée passe la métrique du rectangle.

Idéalement, nous cherchons un planificateur ayant le meilleur taux de couverture pour

le moins de candidats générés afin de réduire le temps nécessaire pour traiter les candidats, ce qui est important dans un scénario interactif.

Classons les générateurs par taux de couverture en % :

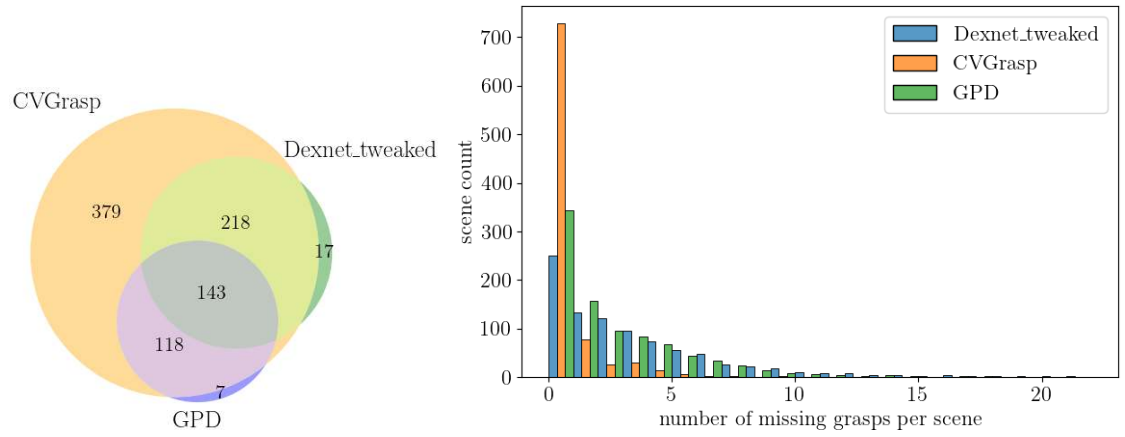
— CVGrasp : 93,44%
— GPD : 64,19%
— Dex-Net modifié : 49,51%
— Dex-Net : 24,736%
— GR-ConvNet Jacquard RGB-D : 11,70%
— GR-ConvNet Jacquard D : 11,27%

CVGrasp a le meilleur taux de couverture, suivi par GPD et Dex-Net modifié, ce dernier nécessitant un temps de planification beaucoup plus long.

Si l'on compare les taux de couverture par rapport au nombre de candidats générés, le classement s'inverse : il devient GR-ConvNet, Dex-Net, GPD, CVGrasp et Dex-Net modifié. Cela met en évidence un compromis entre le taux de couverture et la qualité des saisies générées.

Dans l'objectif d'apprendre comment saisir les objets d'une manière particulière, on préfère les méthodes avec le plus haut taux de couverture, mais cela se fera au détriment d'autres critères recherchés dans les algorithmes de saisie, tels que le nombre de saisies à l'heure (PPH).

Si l'on examine de plus près la répartition du taux de couverture (voir Fig. 6.4), CVGrasp couvre toutes les saisies de 728 scènes. Le diagramme de Venn montre également que Dex-Net modifié trouve plus de saisies manquantes pour CVGrasp que GPD.



(a) (b)

FIGURE 6.4 – (a) Diagramme de Venn : pour chaque scène, on cherche parmi les générateurs ceux ayant le meilleur taux de couverture. Les intersections représentent donc les scènes où plusieurs générateurs sont à égalité. (b) Pour les trois meilleurs algorithmes, on compare le nombre de saisies manquantes par scène.

Test 2 : métrique du rectangle pour chaque générateur. Pour chaque scène, on teste si la meilleure saisie générée par chaque algorithme passe la métrique du rectangle. On remarque que les modèles GR-ConvNet ont le moins d'écart entre leurs résultats à ce test et au test 2, ce qui est dû à la qualité des candidats qu'ils génèrent et retiennent.

GPD et Dex-Net modifié sont les pires à cet égard : parmi les candidats qu'ils génèrent, il existe des saisies adéquates, mais elles ne sont pas retenues par le classement des candidats. CVGrasp obtient un succès sur près de 80% des scènes en assignant des scores uniquement sur des critères géométriques issus des masques de segmentation.

Test 3 : métrique du rectangle pour chaque générateur parmi tous les candidats de chaque scène. Pour chaque scène, on teste si au moins un des candidats de saisies générés passe la métrique du rectangle. Le score des générateurs représente donc le meilleur score que l'on pourrait atteindre avec une parfaite sélection parmi les candidats. Les performances s'alignent avec les résultats du test précédent, mais montrent également qu'il manque au moins un candidat adéquat pour chaque générateurs.

Test 4 : métrique du rectangle pour chaque générateur avec LGPS. On entraîne LGPS de façon similaire à section 3.4 avec toutes les étiquettes disponibles (5 968 étiquettes) (244 scènes sont réservées pour les tests), un VAE entraîné sur les 641 scènes restantes, en substituant le classificateur SVGP par un classificateur SVM [42] avec un noyau quadratique rationnel. Les classificateurs SVM trouvent un hyperplan qui sépare les points de données en différentes classes. Avec plusieurs centaines de labels, les SVM deviennent compétitifs avec le processus gaussien utilisé précédemment. Avec des milliers, ils délivrent des performances égales voire supérieures au SVGP pour un temps d'entraînement plus faible mais un temps d'inférence plus long (voir Tableau 6.4).

Comme remarqué au test 3, avec une meilleure sélection des candidats, GPD et Dex-Net modifié obtiennent de bien meilleurs résultats. Les modèles GR-ConvNet n'ont pas assez de diversité avec leurs candidats pour que l'on observe une amélioration. CVGrasp obtient le meilleur score avec 95% des scènes qui passent la métrique du rectangle, au prix de nombreux candidats à évaluer.

Après avoir analysé les performances des algorithmes de préhension sur la base Cornell, nous avons constaté des résultats intéressants et des compromis entre couverture, qualité des saisies et temps de planification.

CVGrasp a montré le meilleur taux de couverture, avec un taux de réussite de 95% lorsqu'il est associé à LGPS, bien qu'il nécessite l'évaluation de nombreux candidats. GPD et Dex-Net modifié ont également démontré leur potentiel, mais leurs performances pourraient être améliorées grâce à une meilleure sélection des candidats.

Les modèles GR-ConvNet, bien qu'ils aient montré une performance moindre en termes de taux de couverture, ont révélé une qualité supérieure des saisies générées, offrant ainsi un compromis entre ces deux aspects. Cependant, leur faible diversité de candidats limite leur amélioration potentielle lorsqu'ils sont associés à LGPS.

6.3 Comparaisons en simulation

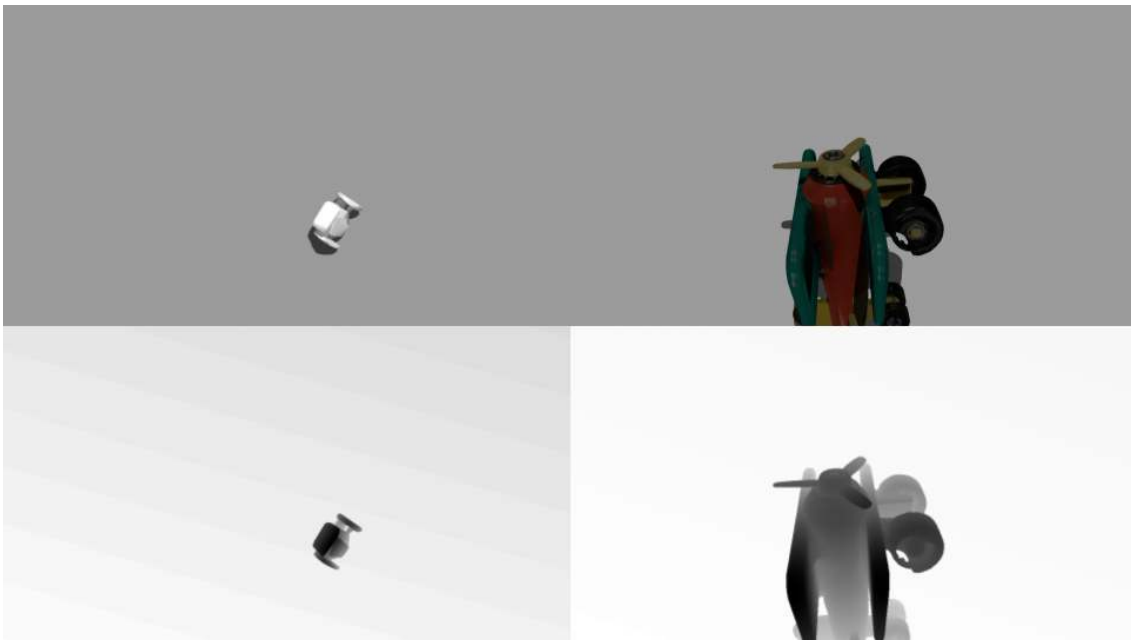


FIGURE 6.5 – Exemples de scènes issues de la simulation, à gauche un objet d'AGOD, à droite un objet de YCB.

La base de données Cornell est composée de scènes dont les objets à saisir se trouvent à plusieurs mètres de la caméra, avec un angle incliné. Cette configuration particulière peut handicaper certains générateurs. Nous réalisons donc des tests en simulation avec deux bases de données, les objets de YCB et ceux d'AGOD (voir sous-section 2.1.2 pour plus d'informations).

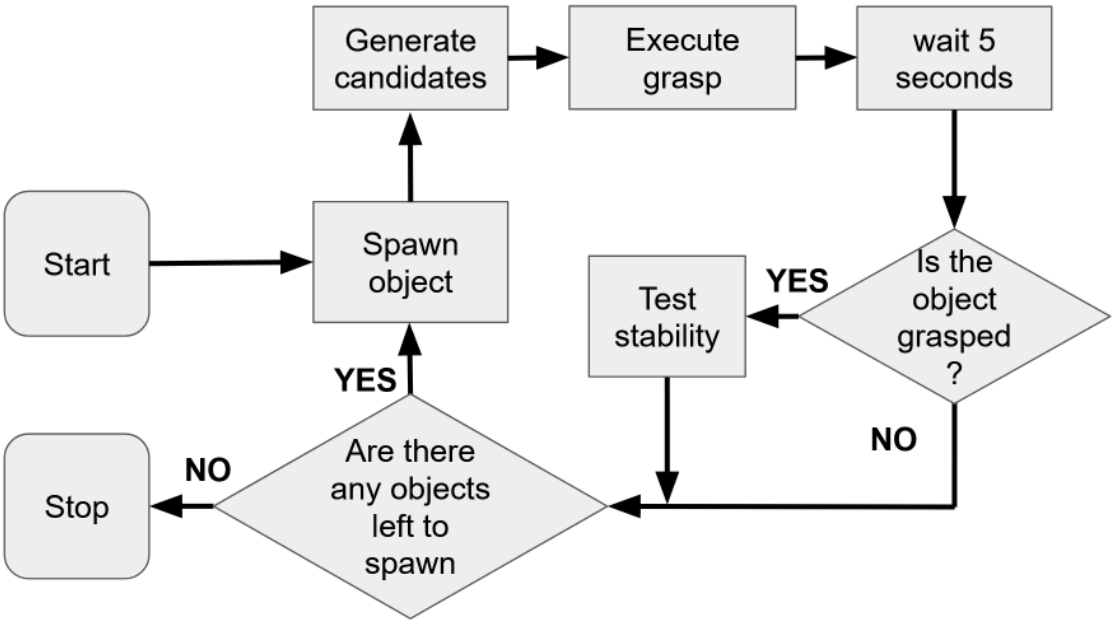


FIGURE 6.6 – Procédure interactive.

Pour cette expérience, tous les algorithmes sont utilisés avec leurs paramètres par défaut. La base de données YCB contient 74 objets, et celle d'AGOD en compte 50. Pour chaque algorithme, on teste 4 positions de chaque objet 5 fois (voir Fig. 6.6), en vérifiant d'abord si l'objet est soulevé, puis, si l'objet est soulevé, s'il passe le test de stabilité précédemment utilisé dans la section 4.2. Les algorithmes sont donc testés sur les mêmes objets, dans les mêmes positions (la caméra sur le préhenseur est située à 70 cm du plan où reposent les objets).

TABLE 6.2 – Résultats sur les objets YCB.

generator name	grasp success (%)	stability (%)
CVGrasp	63,46	33,36
GR-ConvNet Jacquard rbgd	61,22	23,86
Dex-Net	52,34	26,17
GR-ConvNet Jacquard d	48,88	25,76
GPD	45,56	20,75
GR-ConvNet Cornell rbgd	42,91	23,80

TABLE 6.3 – Résultats sur les objets AGOD

generator name	grasp success (%)	stability (%)
CVGrasp	73,2	35,1
GR-ConvNet Jacquard rbgd	63,5	21,8
GR-ConvNet Cornell rbgd	61,7	20,9
GPD	53,1	15,7
Dex-Net	50,1	19,6
GR-ConvNet Jacquard d	49,7	22,1

Au vu des résultats (voir Tab. 6.2 et 6.3), les modèles GR-ConvNet généralisent beaucoup mieux dans cette configuration que pour les scènes de Cornell. En général, on suppose que utiliser seulement les données de profondeur produit une meilleure généralisation, mais cela ne semble pas être le cas ici, le modèle entraîné sur les données RGB-D de Jacquard performant mieux que Dex-Net et GPD.

CVGrasp est le meilleur sur les deux bases de données, possiblement grâce à la qualité de la segmentation des images de profondeur générées dans Gazebo, dépourvues du bruit (Fig 6.5) présent dans les images réelles.

On observe une grande variabilité de la difficulté des objets et de la réussite des algorithmes, comme le montrent les graphiques présentés dans la Fig. 6.7 et la Fig. 6.8. Il est intéressant de noter qu'un même objet peut avoir un taux de réussite proche de 100% pour une méthode, mais un taux très faible pour une autre. En outre, six objets YCB semblent être inaccessibles dans la simulation en raison de leur taille ou de leur poids, tandis que six autres objets ont un taux de réussite inférieur à 20%. Les objets AGOD présentent une bonne diversité de taux de réussite, tout en restant saisissables.

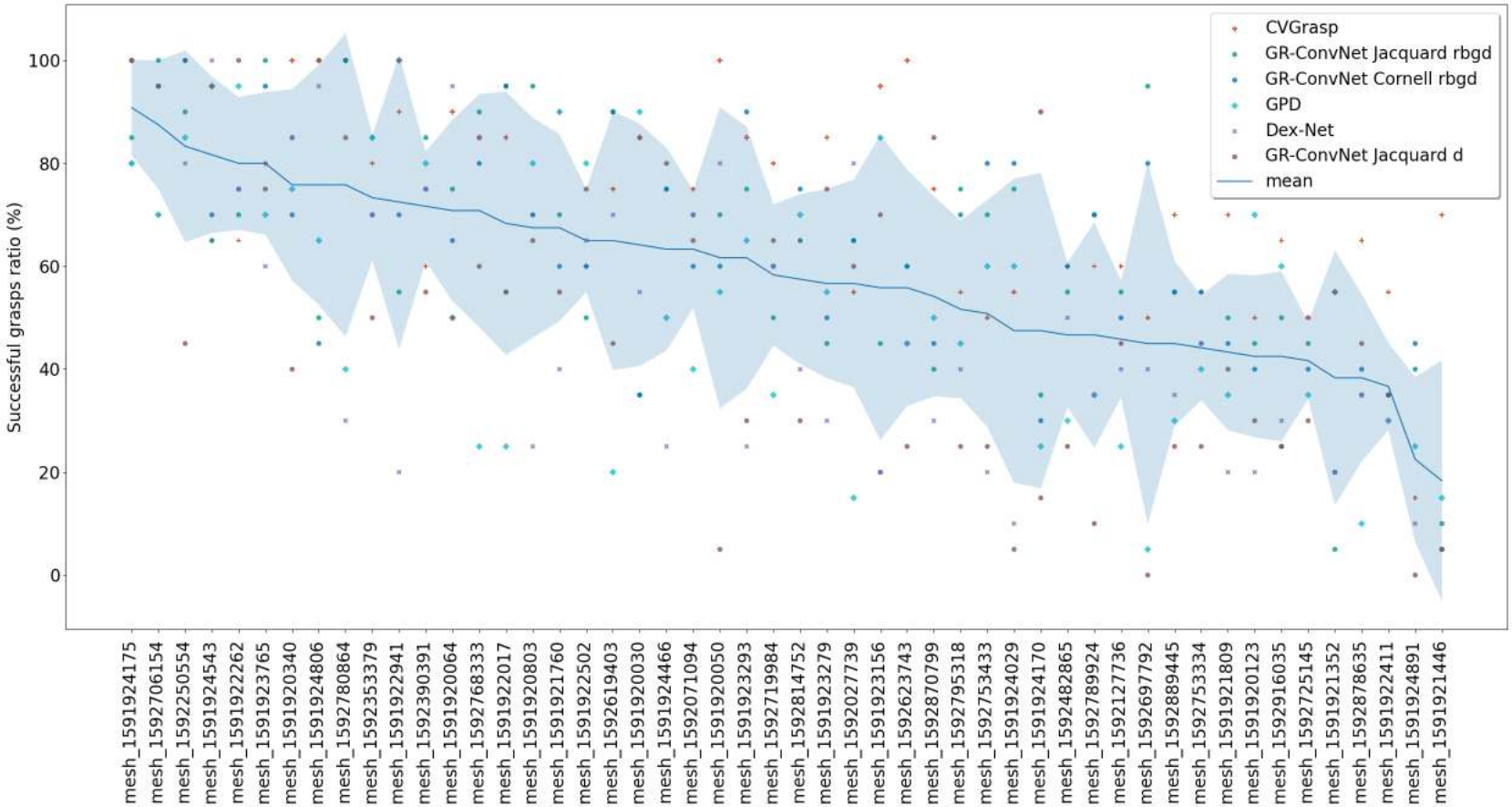


FIGURE 6.7 – Variation du taux de succès de préhension des objets de l'ensemble de données AGOD.

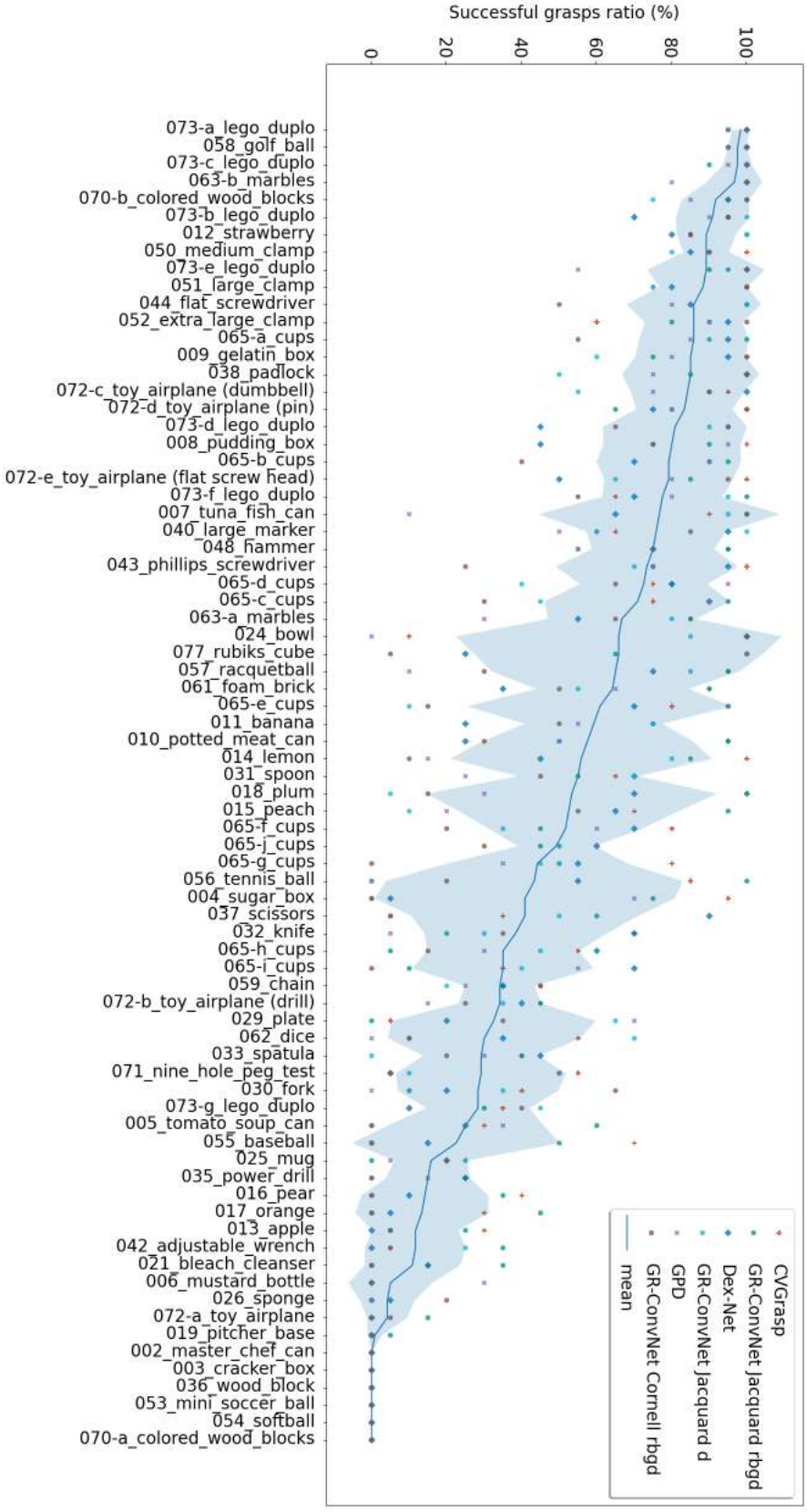


FIGURE 6.8 – Variation du taux de succès de préhension des objets de l'ensemble de données YCB.

6.4 Expérience avec le robot

Nous évaluons et comparons les performances de quatre algorithmes de préhension différents : GPD, Dex-Net, GR-ConvNet et CVGrasp. L'objectif de l'expérience est de déterminer quel algorithme est le plus efficace pour saisir une variété d'objets, et si chaque algorithme peut soulever un objet avec succès tout en atteignant un taux de couverture de saisie satisfaisant.

L'expérience teste chaque algorithme sur six objets différents, chacun ayant une manière spécifique d'être saisi (Fig. 6.9).

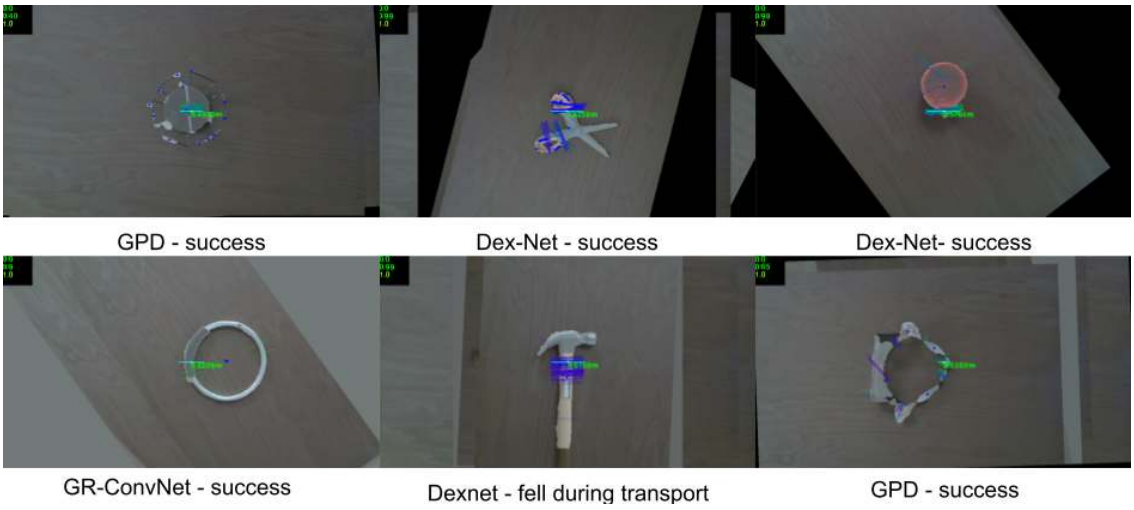


FIGURE 6.9 – Les objets utilisés lors de l'expérience et des exemples de saisies souhaitées générées par les différents algorithmes. Le marteau, malgré une saisie conforme aux attentes de l'expert, est tombé.

GPD : 4 préhensions réussies sur 6 objets, avec 2 objets sur 6 présentant une diversité suffisante de candidats et 2 saisies correspondant à la manière spécifique d'être saisi.
Dex-Net : 4 préhensions réussies sur 6 objets, avec 3 objets sur 6 présentant une diversité suffisante de candidats et 3 saisies correspondant à la manière spécifique d'être saisi.
CVgrasp : 3 préhensions réussies sur 6 objets, avec 6 objets sur 6 présentant une diversité suffisante de candidats et 3 saisies correspondant à la manière spécifique d'être saisi.
GR-ConvNet jacquard RGB-D : 3 préhensions réussies sur 6 objets, avec 4 objets sur 6 présentant une diversité suffisante de candidats et 1 saisies correspondant à la manière spécifique d'être saisi.
(GPD+Dex-Net+GR-ConvNet) + LGPS (4 labels par objets) : 5 préhensions réussies sur 6 objets, avec 6 objets sur 6 présentant une diversité suffisante de candidats et 5 saisies correspondant à la manière spécifique d'être saisi.

Les résultats de l'expérience montrent que, par défaut, les algorithmes ne parviennent pas majoritairement à saisir les objets de la manière préférée par l'expert. Toutefois, en combinant les algorithmes GPD, Dex-Net et GR-ConvNet, nous obtenons un taux de couverture comparable à celui de CVgrasp pour ces objets spécifiques. Il est important de noter que cette comparaison ne s'applique pas à la base de données de Cornell (voir Fig. 6.4 pour plus de détails). En intégrant l'approche LGPS, nous parvenons à apprendre

et à reproduire les préférences de saisie de l'expert, ce qui améliore la qualité des prises générées par les algorithmes combinés.

L'expérience menée avec le robot pour évaluer et comparer les performances des quatre algorithmes de préhension (GPD, Dex-Net, GR-ConvNet et CVGrasp) a révélé des résultats intéressants. Par défaut, aucun des algorithmes ne parvient majoritairement à saisir les objets de la manière préférée par l'expert. Cependant, en combinant les algorithmes GPD, Dex-Net et GR-ConvNet, nous avons pu obtenir un taux de couverture comparable à celui de CVGrasp pour ces objets spécifiques, bien que cette comparaison ne s'applique pas à la base de données de Cornell.

Avec l'approche LGPS il est possible d'apprendre et de reproduire les préférences de saisie de l'expert, améliorant ainsi la qualité des prises générées par les algorithmes combinés. Cette étude souligne l'importance de l'exploration de méthodes hybrides et de l'adaptation des algorithmes existants pour améliorer leur performance sur des tâches spécifiques. Les travaux futurs pourraient se concentrer sur le développement d'algorithmes plus robustes et sur l'exploration de techniques d'apprentissage permettant de personnaliser davantage les préférences de saisie en fonction des experts et des scénarios d'application.

6.5 Temps de génération des candidats pour l'interaction

Dans le contexte d'une expérience interactive avec des étiquettes ajoutées graduellement à la base d'entraînement suite aux tentatives du robot, l'expert doit patienter durant la génération des candidats et l'exécution de la saisie, prenant en moyenne 30 secondes. Comme le temps nécessaire aux mouvements du robot est indépendant de la planification de la saisie, nous nous intéressons au temps requis pour générer et traiter les candidats avec LGPS.

Les temps d'exécution des algorithmes (voir Tableau 6.1) pour la planification sont similaires à ceux rapportés dans les publications :

— Dex-Net : 2,5 secondes (processeur 3,4 GHz Intel Core i7-6700 quad-core - carte graphique NVIDIA TITAN Xp GPU).
— GR-ConvNet : 0,02 seconde (processeur Intel Core i7-7800X CPU à 3,50 GHz - carte graphique NVIDIA GeForce GTX 1080 Ti GPU avec CUDA 10).
— GPD : entre 1 et 8 secondes (processeur Intel Core i7-4770K Haswell Quad-Core 3,5 GHz, 32 Go de RAM - carte graphique NVIDIA GTX 970 GPU).

Dans notre implémentation, il faut néanmoins ajouter le temps nécessaire pour créer la requête (entre 1 et 1,5 seconde) pour les nœuds ROS générant les candidats (la même requête peut être utilisée lorsqu'il y a plusieurs générateurs).

La segmentation de la scène pour isoler les objets est requise uniquement pour CVGrasp, et cette opération prend entre 0,5 et 1 seconde (section A.6).

TABLE 6.4 – Temps de traitement pour différentes étapes.

step	processing time (s)
5698 grasps training SVM	2,73
5698 grasps training SVGP	10
3588 grasps training SVR	0,54
1000 grasps patch encoding RGB	1,36
1000 grasps patch encoding D	1,31
1000 grasps patch encoding SN	2,14
1000 grasps patch encoding RGB-D	2,46
1000 grasps patch encoding RGB-SN	3,36
1000 grasps patch encoding D-SN	2,38
1000 grasps patch encoding RGB-D-SN	3,8
1000 grasps VAE encoding	3,4
1000 grasps classification SVM	0.21
1000 grasps classification SVGP	0.03
1000 grasps regression SVR	0.02

Le temps nécessaire pour traiter les candidats est directement lié à leur nombre. À cet égard, CVGrasp est le moins performant, surtout si la scène comporte plusieurs objets à saisir. Sur la base de données Cornell, le nombre moyen de candidats par scène est de 692 (Tableau 6.1).

L'extraction des patchs (sous-section 3.3.2) est l'opération la plus longue. Utiliser uniquement les données RGB ou de profondeur est une option pour réduire ce temps. Sur la base de données de Cornell, l'utilisation de données RGB uniquement diminue les performances d'environ 5%.

Une amélioration mise en place consiste à stocker les représentations dans l'espace latent des étiquettes d'entraînement, afin d'éviter de les régénérer à chaque nouvel entraînement entre deux tentatives.

Le temps total estimé est de 50 secondes par saisie dont 30 secondes d'exécution par le robot. En comparaison, un opérateur humain moyen effectue 56,62 saisies par heure, soit environ 63,58 secondes par saisie (voir section 5.4). Pour obtenir des performances comparable, cela nous laisse une quinzaine de secondes par correction dans le cas où l'utilisateur devrait intervenir, ce qui est suffisant.

6.6 Synthèse

Il est difficile de déterminer un "meilleur" générateur parmi CVGrasp, GPD, Dex-Net et GR-ConvNet, car leur performance et leur efficacité dépendent des conditions et des exigences spécifiques de l'application.

— Rapidité : GR-ConvNet est de loin le plus rapide.
— Taux de couverture des saisies : CVGrasp a eu le meilleur taux de couverture, suivi par GPD et DexNet. GR-ConvNet est le pire.
— Qualité de la saisie : CVGrasp a obtenu les meilleurs résultats en simulation et sur la base de données Cornell. GPD offre des saisies en 6DOF, ce qui est un grand

avantage, et performe bien sur Cornell. GR-ConvNet entraîné sur Jacquard a des difficultés sur la base de données Cornell mais est plus performant en simulation, où l'angle et la distance des scènes traitées sont plus similaires à sa base de données d'entraînement, laissant penser qu'il y a des difficultés à généraliser.

CVGrasp est performant pour des scènes simples, avec peu d'objets et sans besoin de saisies 6DOF. Ses atouts sont un taux de couverture ajustable et exhaustif, ainsi qu'une planification assez rapide.

Dans les autres cas, comme lorsqu'on doit vider un bac, l'expérience avec le robot (section 6.4) a montré que combiner plusieurs générateurs est un compromis intéressant pour générer de nombreux candidats assez divers. Sur la base de données Cornell, on peut observer sur la Figure 6.4 que les saisies trouvées par GPD et DexNet ne se recouvrent pas, ce qui est intéressant. Les modèles GR-ConvNet sont très rapides et génèrent peu de candidats, mais de haute qualité, ce qui est donc peu coûteux à utiliser avec LGPS.

6.7 Apprentissage actif

L'apprentissage actif [165] est une stratégie d'apprentissage automatique dans laquelle un algorithme peut interroger de manière interactive un utilisateur ou toute autre source d'information afin d'attribuer des étiquettes aux nouveaux points de données en fonction des résultats souhaités. Cette source d'information interactive est souvent désignée comme étant un "enseignant" ou un "oracle".

Il existe des situations où les données non étiquetées sont abondantes (les candidats de préhension générés par les générateurs), mais où l'étiquetage manuel est coûteux (quel préhension correspond aux attentes de l'utilisateur). Dans de tels scénarios, les algorithmes d'apprentissage peuvent interroger activement l'utilisateur ou l'enseignant pour obtenir des étiquettes. Ce type d'apprentissage supervisé itératif est appelé apprentissage actif. Il cherche à optimiser à la fois la précision du modèle et l'efficacité de l'étiquetage des données. En se concentrant sur une sélection stratégique des données à annoter, l'apprentissage actif vise à atteindre un niveau élevé de performance avec un volume réduit de données. L'apprentissage actif trouve son utilité particulièrement dans les situations où l'annotation des données est coûteuse ou complexe.

La méthode initiale d'apprentissage actif consiste à entraîner un modèle sur une petite quantité de données déjà annotées. Ensuite, des points de données supplémentaires sont sélectionnés pour l'annotation, basée sur leur potentiel d'amélioration des performances du modèle. La méthode d'interrogation la plus courante [165] consiste à identifier ces points de données en fonction de l'incertitude du modèle à leur égard ou de leur capacité à influencer de manière significative la précision globale du modèle. Une fois que ces données supplémentaires ont été sélectionnées et annotées, le modèle est réentraîné sur l'ensemble de données enrichi. Cette procédure est répétée jusqu'à l'obtention du niveau de performance désiré.

6.7.1 Test sur la base de données Cornell

LGPS peut utiliser un classifieur à processus gaussien binaire qui génère la variance vis-à-vis de ses prédictions. Comparons différentes stratégies sur la base de données Cornell

pour sélectionner le prochain échantillon à ajouter à la base d'entraînement, **parmi les saisies étiquetées disponibles dans la base de données**.

Nous expérimentons quatre stratégies d'apprentissage actif, spécialement conçues pour identifier les échantillons les plus informatifs à étiqueter :

— Stratégie 1 - Échantillonnage aléatoire : Cette stratégie ne fait pas de supposition sur l'information portée par les échantillons et sélectionne simplement un échantillon de manière aléatoire pour l'étiquetage.

— Stratégie 2 - Maximisation de la variance : Cette stratégie cherche à sélectionner les échantillons ayant la plus grande variance dans les prédictions, c'est-à-dire le plus grand degré d'incertitude sur l'étiquette attendue. La variance est prise entre 0 et 0,25 et est multipliée par 4 pour obtenir le score :

$$score = 4var$$

— Stratégie 3 - Maximisation de l'incertitude : Cette stratégie vise à choisir les échantillons pour lesquels l'incertitude de la prédiction est la plus grande, c'est-à-dire les échantillons dont la probabilité prédite est proche de 0,5. Le score est déterminé par :

$$score = 1 - 2\left|\frac{1}{2} - mean\right|$$

— Stratégie 4 - Maximisation de l'incertitude et de la variance : Cette stratégie est une combinaison des Stratégies 2 et 3. Elle cherche à sélectionner des échantillons qui ont à la fois une grande incertitude de prédiction et une grande variance dans les prédictions. Le score est calculé en faisant la moyenne des scores obtenus selon la Stratégie 2 et la Stratégie 3 :

$$score = ((1 - 2\left|\frac{1}{2} - mean\right|) + 4var)/2$$

Ces stratégies reflètent différentes hypothèses sur quels échantillons pourraient être les plus informatifs pour l'apprentissage du modèle. Elles cherchent à maximiser l'efficacité de l'étiquetage en concentrant les efforts sur les échantillons qui sont susceptibles d'améliorer le plus la performance du modèle.

Nous évaluons la performance de chaque stratégie en mesurant le pourcentage de scènes réussissant la métrique du rectangle lorsqu'elles sont entraînées avec un nombre croissant d'échantillons étiquetés, jusqu'à un maximum de 231. Pour chaque stratégie, nous entraînons 5 modèles différents.

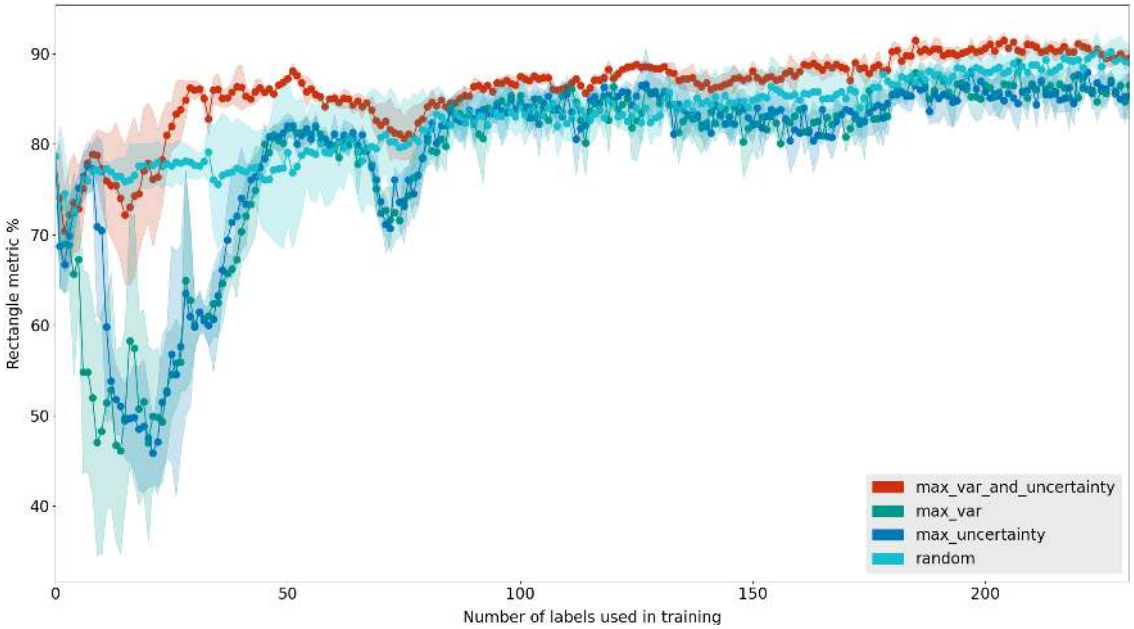


FIGURE 6.10 – L'axe x correspond au nombre d'étiquettes utilisées pour l'apprentissage (641 scènes) et l'axe y à la métrique du rectangle en pourcentage du nombre de scènes dans la base de données de test (244 scènes). À 0 étiquette, le score des saisies est celui du générateur CVGrasp.

Nos résultats (Figure 6.10) montrent que la stratégie 4, qui combine à la fois l'incertitude et la variance dans les prédictions, semble être la méthode la plus efficace pour sélectionner le prochain échantillon à étiqueter. L'échantillonnage aléatoire (stratégie 1) s'avère être supérieure aux autres méthodes basées uniquement sur l'incertitude ou la variance. Cependant, toutes les stratégies ont des performances similaires lorsqu'elles sont entraînées avec plus de 100 échantillons étiquetés. Dans ce régime, les différences entre les stratégies sont minimes et généralement comprises dans une plage de quelques points de pourcentage.

Les conditions de cette expérience sont exceptionnelles car la sélection de la prochaine donnée d'entraînement est effectuée à partir d'un ensemble de préhensions étiquetées préalablement par un expert et disponibles dans une base de données. Par conséquent, ces saisies représentent déjà des exemples pertinents de bonnes et de mauvaises saisies.

6.7.2 Test de l'apprentissage actif en simulation

Lorsqu'une nouvelle base de données est utilisée, il est possible de sélectionner la préhension à étiqueter parmi une série de candidats générés pour une scène spécifique. Si un expert est disponible, celui-ci peut évaluer la qualité de la préhension proposée avant de l'exécuter et corriger la saisie à effectuer si nécessaire. À l'aide de l'interface graphique utilisateur (GUI), l'expert peut signaler que la saisie suggérée par le classificateur est inappropriée et choisir parmi les candidats une alternative plus adéquate. Sinon, la qualité de la préhension peut être évaluée en fonction de son succès lors de l'exécution, comme le montre la Figure 6.11.

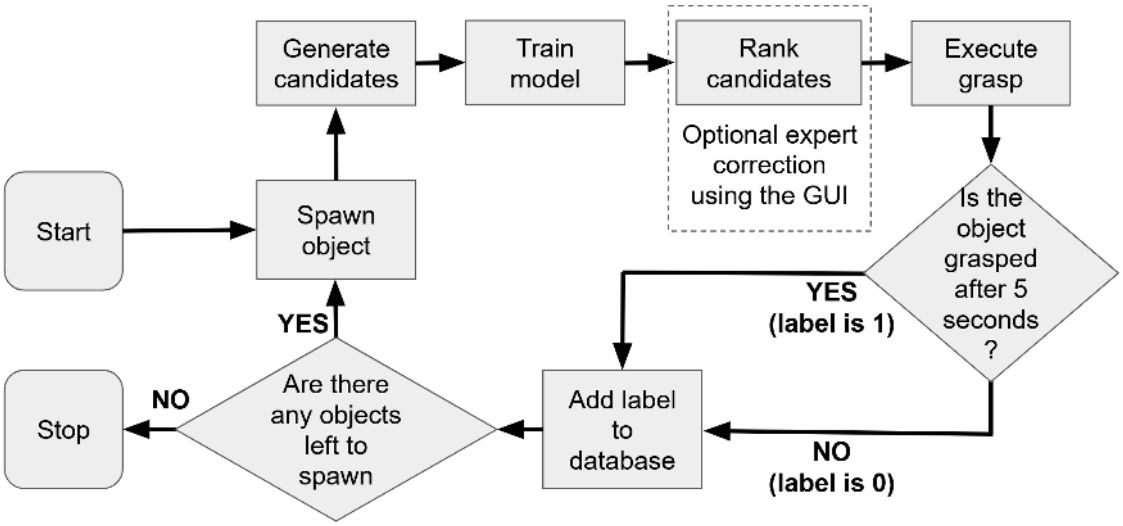


FIGURE 6.11 – Procédure interactive en simulation.

Nous réutilisons les bases de données utilisées dans section 6.3, mais cette fois-ci en ajoutant LGPS avec CVGrasp comme générateur de préhension. Au début, le classificateur de LGPS n'est pas entraîné, et donc la meilleure saisie de CVGrasp est utilisée. Après chaque tentative, la saisie effectuée est ajoutée à la base de données d'entraînement de LGPS. Lorsqu'un exemple positif et négatif sont présents dans la base de données, le classificateur peut évaluer les saisies après la génération afin de sélectionner la saisie à tester parmi les candidats. Pour les 4 premières saisies par objets (1ère colonne des tableaux 6.6 et 6.5), plusieurs stratégies sont testées :

— vanilla - Standard : la saisie préférée par le classificateur est exécutée.

— max_var - Variance maximale : en utilisant les échantillons avec la plus grande variance dans les prédictions (entre 0 et 0,25).

$$score = 4var$$

— max_uncertainty - Incertitude maximale : en utilisant des échantillons avec la plus grande incertitude de prédiction (c'est-à-dire, une probabilité proche de 0,5).

$$score = 1 - 2\left|\frac{1}{2} - mean\right|$$

— max_var_and_uncertainty - Incertitude et variance maximales : en utilisant les échantillons à la fois avec une grande incertitude de prédiction et une grande variance dans les prédictions.

$$score = ((1 - 2\left|\frac{1}{2} - mean\right|) + 4var)/2$$

Pour la suite (2ème colonne des tableaux 6.6 et 6.5), la saisie préférée par le classificateur est exécutée. L'évaluation en simulation de ces stratégies permettra de déterminer laquelle est la plus efficace pour améliorer les performances de l'algorithme de préhension lorsqu'il est confronté à de nouvelles scènes et objets.

TABLE 6.5 – Résultats moyens sur les objets AGOD, chaque stratégie étant testée 3 fois.

strategy	grasp success rate (%) from grasp 1 to 200	grasp success rate (%) from grasp 201 to 400
max_var_and_uncertainty	54,833	87,667
max_var	53,0	**88,833**
vanillla	**84,167**	87,0
max_uncertainty	47.5	86.333

TABLE 6.6 – Résultats moyens sur les objets YCB, chaque stratégie étant testée 3 fois.

strategy	grasp success rate (%) from grasp 1 to 295	grasp success rate (%) from grasp 296 to 590
max_var_and_uncertainty	50,395	72,881
max_var	48,135	74,689
vanillla	**73,446**	75,028
max_uncertainty	45,31	**76,723**

En tenant compte des conclusions de l'expérience précédente (sous-section 6.7.1), il est important de considérer le contexte dans lequel les stratégies ont été évaluées. Dans l'expérience mentionnée, la sélection des données d'entraînement était basée sur un ensemble de préhensions étiquetées préalablement par un expert, ce qui signifie que ces préhensions constituaient déjà des exemples pertinents de bonnes et de mauvaises saisies.

La stratégie 4, qui combine à la fois l'incertitude et la variance, a été identifiée comme la méthode la plus efficace pour sélectionner les échantillons à étiqueter dans l'expérience précédente. Toutefois, dans le contexte des résultats présentés dans les tableaux 6.6 et 6.5, on observe que malgré les différentes stratégies, les performances du système de préhension restent similaires.

Il est possible que la différence de performance entre les stratégies découle des différences dans les conditions d'expérimentation. Par conséquent, il semble que les stratégies qui fonctionnent bien dans un contexte d'apprentissage actif avec une base de données pré-étiquetées ne soient pas nécessairement les meilleures pour le scénario interactif.

L'ajout de LGPS au processus a permis d'observer une amélioration notable des performances de préhension par rapport à l'utilisation de CVGrasp seul. Comme mentionné précédemment (voir section 6.3), CVGrasp obtenait un taux de succès de 63,46% sur la base de données YCB et un taux de succès de 73,2% sur la base de données AGOD. En intégrant LGPS dans le processus, les taux de succès ont augmenté de 10 à 15%, grâce à l'ajustement interactif de la sélection des préhensions, sans même l'intervention de l'expert.

En conclusion, il est essentiel de prendre en compte le contexte dans lequel les différentes stratégies sont évaluées, car celles qui sont efficaces dans un contexte d'apprentissage actif avec des données pré-étiquetées ne sont pas nécessairement les meilleures pour un scénario interactif. L'intégration de LGPS avec CVGrasp a montré une amélioration significative des performances, suggérant que l'approche interactive peut être prometteuse pour l'apprentissage actif et l'amélioration des algorithmes de préhension.

6.7.3 Test de l'apprentissage interactif en simulation

Dans le but d'intégrer l'humain dans la boucle, la stratégie vanilla semble être la plus appropriée, car les performances lors des premières saisies restent proches des performances finales. Dans cette seconde expérience, l'expert intervient lors des quatre premières saisies par objet.

TABLE 6.7 – Résultats sur les objets AGOD avec correction.

grasp success rate (%) grasp 1 to 200 with expert correction	100
number of intervention	36 intervention (18% of grasps)
grasp success rate (%) grasp 201 to 400	89
grasp success rate (%) grasp 401 to 695 **on YCB dataset**	72,542

TABLE 6.8 – Résultats sur les objets YCB avec correction.

grasp success rate (%) grasp 1 to 295 with expert correction	86,779
number of intervention	76 intervention (25,7% of grasps)
grasp success rate (%) grasp 296 to 590	77,966
grasp success rate (%) grasp 591 to 790 **on AGOD dataset**	89,5

Les résultats obtenus pour les objets AGOD et YCB avec la correction de l'expert sont présentés dans les tableaux 6.7 et 6.8, respectivement. Dans le cas des objets AGOD, avec l'intervention de l'expert, un taux de réussite de 100% a été atteint pour les 200 premières saisies. L'expert a dû intervenir 36 fois, ce qui représente 18% des saisies. Pour les saisies 201 à 400, le taux de réussite était de 89%, ce qui est supérieur au meilleur cas sans intervention de 88,833. Lors de l'application de l'algorithme sur l'ensemble de données YCB (saisies 401 à 695), un taux de réussite de 72,54% a été obtenu.

Pour les objets YCB, avec l'intervention de l'expert, un taux de réussite de 86,78% a été atteint pour les 295 premières saisies. L'expert a dû intervenir 76 fois, ce qui représente 25,7% des saisies. Pour les saisies 296 à 590, le taux de réussite était de 77,97%, ce qui est supérieur au meilleur cas sans intervention de 76,723%. Lors de l'application de l'algorithme sur l'ensemble de données AGOD (saisies 591 à 790), un taux de réussite de 89,5% a été obtenu.

Ces résultats montrent que l'intervention humaine, même limitée (18% et 25,7% des saisies), améliore les performances du système de préhension. De plus, une fois que le système a appris des interventions de l'expert, les performances restent supérieures à celles obtenues sans l'intégration de l'expertise humaine. Cela démontre l'efficacité de l'approche interactive et de l'intégration de l'expertise humaine dans le processus d'apprentissage.

La capacité de généralisation de l'algorithme semble également prometteuse. Lorsque pré-entraîné sur l'ensemble de données YCB, l'algorithme a obtenu un taux de succès de 89,5% sur l'ensemble de données AGOD, ce qui est même supérieur au taux de réussite de 89% obtenu pour les saisies 201 à 400 réalisées après l'intervention de l'expert. Cependant, les performances de généralisation sur l'ensemble de données YCB après le pré-entraînement sur les objets de AGOD sont légèrement inférieures (72,54%), bien que toujours supérieures au taux de succès de 63,46% obtenu sans LGPS.

Il est important de noter que ces expériences ont été réalisées avec un seul expert.

Par conséquent, les résultats pourraient varier en fonction de l'expertise et de la connaissance spécifiques de chaque individu. Il serait intéressant d'étendre cette étude à plusieurs utilisateurs pour évaluer la variabilité des performances et la robustesse de l'approche, d'autant plus que les travaux précédents ont montré de grandes différences de performance entre différents experts (chapitre 5).

En conclusion, bien que les résultats soient encourageants, il est nécessaire de poursuivre les recherches et d'élargir les expériences à un plus grand nombre d'utilisateurs et dans des conditions difficiles pour évaluer la robustesse de la méthode dans divers contextes et niveaux d'expertise.

6.8 Conclusions

Au cours de ce chapitre, nous avons comparé diverses méthodes de génération de préhensions robotiques, CVGrasp, GPD, Dex-Net et GR-ConvNet. La performance et l'efficacité de ces approches dépendent des conditions spécifiques et des exigences de l'application. GR-ConvNet se démarque par sa rapidité, tandis que CVGrasp offre le meilleur taux de couverture des saisies et la meilleure qualité de saisie dans nos expériences. De plus, GPD présente l'avantage significatif de proposer des saisies en 6DOF. CVGrasp est suffisamment performant lorsque la segmentation de la scène est simple. Cependant, dans des situations plus complexes, l'utilisation combinée de plusieurs générateurs permet de générer des candidats diversifiés et d'améliorer la performance.

Concernant l'apprentissage actif, nos résultats indiquent que la stratégie combinant l'incertitude et la variance est la plus efficace pour sélectionner les échantillons à étiqueter. Toutefois, il est important de souligner que les conditions de cette expérience sont exceptionnelles, puisque la sélection de la prochaine donnée d'entraînement est effectuée à partir d'un ensemble de préhensions déjà étiquetées par un expert et disponibles dans une base de données. Par conséquent, ces saisies représentent déjà des exemples pertinents de bonnes et mauvaises saisies. De plus, les différences entre les stratégies deviennent minimes lorsque l'entraînement est effectué avec plus de 100 échantillons étiquetés. Dans les tests en simulation, les différentes stratégies sont similaires à ceux obtenus en sélectionnant la saisie préférée par le classificateur.

L'intégration interactive de LGPS améliore significativement les performances de préhension (environ +15% de taux de succès) en ajustant la sélection des préhensions après chaque essai, même sans intervention de l'expert (environ +12% de taux de succès). L'intervention de l'expert humain permet d'améliorer encore davantage les performances de préhension, démontrant ainsi l'efficacité de l'approche interactive et de l'intégration de l'expertise humaine dans le processus d'apprentissage.

Les résultats en matière de généralisation sont prometteurs, avec un taux de réussite élevé lorsque l'algorithme est pré-entraîné sur un ensemble de données et testé sur un autre. Cependant, il est essentiel d'élargir les expériences à un plus grand nombre d'experts et à des conditions plus difficiles pour évaluer pleinement la robustesse de la méthode dans divers contextes et avec différents niveaux d'expertise.

En conclusion, ce chapitre met en lumière les avantages et les limites des différentes approches de génération de préhensions robotiques, ainsi que l'amélioration apportée par l'intégration de l'expertise humaine dans le processus. Les résultats obtenus sont encou-

rageants, mais des recherches supplémentaires sont nécessaires pour étudier la robustesse et l'adaptabilité de ces méthodes dans des scénarios divers et complexes.

Chapitre 7

Discussion et perspectives

7.1 Contributions

Dans l'ensemble, cette thèse a développé une approche efficace en termes de données et adaptable pour apprendre des saisies particulières ou générales. Nous avons exploré diverses modalités sensorielles et leur intégration dans un cadre unifié d'apprentissage automatique. De plus, nous avons encouragé la mise en place de tests standardisés et la création d'ensembles d'objets d'évaluation fixes pour faciliter les comparaisons.

L'étude des performances humaines dans la tâche de saisie d'objets en vrac a offert des perspectives intéressantes pour le développement et l'évaluation des systèmes robotiques de manipulation d'objets. L'examen des erreurs et des échecs humains permet d'anticiper les défis auxquels les robots pourraient être confrontés et d'optimiser leur fonctionnement en conséquence. Les performances des utilisateurs fournissent également une base de comparaison utile pour évaluer les algorithmes.

Nous avons comparé plusieurs générateurs de saisies, tels que CVGrasp, GPD, DexNet et GR-ConvNet, et nous avons constaté qu'il est difficile de déterminer un "meilleur" générateur, car leurs performances et leur efficacité dépendent des conditions et des exigences spécifiques à chaque application. La combinaison de plusieurs générateurs semble constituer une solution intéressante pour générer un grand nombre de candidats diversifiés lorsque CVGrasp n'est pas adéquat.

L'utilisation du capteur DIGIT a permis d'améliorer les performances de la préhension robotique en intégrant des informations tactiles. Le retour d'information de la zone DIGIT peut fournir des indications sur la qualité de la prise, permettant d'ajuster la stratégie de saisie et d'apprendre à prédire de meilleures hauteurs de saisie. Les capteurs DIGIT peuvent être utilisés pour la collecte automatique de données, bien qu'un défi subsiste : tester la profondeur automatiquement déplace l'objet, ce qui peut entraîner des problèmes lors de la saisie si l'objet n'est pas stable.

7.2 Amélioration des propositions

Cependant, de nombreuses possibilités d'amélioration et de développement futur subsistent.

Le préhenseur que nous avons utilisé ne permet pas de contrôler la force exercée lors

de la saisie, ce qui pourrait être appris de l'humain, à l'aide des DIGITS ou d'autres capteurs.

LGPS ne permet pas de saisie en temps réel, ce qui signifie également que l'objet peut se déplacer entre le moment où la décision est prise et l'action effectuée. Des implémentations plus rapides pourraient atténuer cette effet, mais la séparation entre génération et classification, inhérente à notre approche, restera un facteur limitant.

Bien que notre implémentation permette de collecter la position finale de la saisie via le GUI ou la manette d'XBOX 360, la séparation du générateur et du classificateur limite également l'apprentissage de manipulations complexes.

Néanmoins, notre approche devrait être compatible avec de nouveaux générateurs de saisie, méthodes de segmentation ou de reconnaissance d'objets, ce qui permet une intégration facile et une amélioration continue des performances. La compatibilité avec les algorithmes 6DOF est assurée, car le classificateur de patchs évalue la zone générale où la saisie sera effectuée en fonction des besoins de l'utilisateur.

Cependant, cette représentation par des patchs localisés de taille limitée peut entraîner des erreurs de classification si la cause de l'échec ne se trouve pas dans la zone de saisie. Par exemple, un marteau peut être bloqué à une extrémité et non à l'autre, et le classificateur pourrait considérer la saisie comme correcte. Un autre cas envisageable est celui d'objets de formes similaires mais avec des préférences différentes qui peuvent poser problème. Par exemple, saisir tous les couverts par leur manche, sauf les couteaux qui doivent être saisis par leur lame. Si les manches sont similaires, la prise sur eux serait considérée comme bonne.

L'utilisation de représentations plus complexes (ex. PointNet VAE pour toute la scène) ou de patchs plus grands pourrait améliorer les résultats, mais rallongerait le temps de traitement et pourrait compromettre l'efficacité des classificateurs en raison d'une représentation latente plus grande.

La scalabilité de LGPS n'a pas été testée avec de très grands ensembles de données, celle-ci pourrait également être limitée par les classificateurs efficaces en données.

Une piste explorable : nous avons montré que le pipeline de préhension peut être modifié pour reconnaître des objets ou utiliser d'autres classificateurs existants. Nous pourrions entraîner plusieurs classificateurs en fonction des classes d'objets à saisir. Lorsqu'une saisie est testée, le classificateur approprié peut la classer. Cela pourrait aider à la scalabilité et contourner les problèmes liés aux objets de formes similaires. Nous avons testé cette approche avec la base de données Cornell, mais comme celle-ci est davantage axée sur la préhension en général, l'utilisation d'un seul classificateur était plus efficace.

Les performances avec différents utilisateurs et l'aspect humain dans la boucle n'ont pas été suffisamment testés, bien que les premiers tests semblent prometteurs. Les algorithmes qui fonctionnent avec des objets isolés fonctionnent généralement avec des tas d'objets. Cependant, nous avons testé le scénario du vidage de bac en remplaçant l'humain par GPD, DexNet et GR-ConvNet. Nous n'avons pas eu le temps de formaliser ces tests, mais les algorithmes se sont tous les trois retrouvés bloqués, exécutant plusieurs fois la même saisie qui échouait, sans bouger suffisamment les objets dans le bac pour sortir de cette boucle. La version interactive ne devrait pas se bloquer dans ce scénario, car l'humain devrait pouvoir la débloquer en fournissant une saisie adéquate. Idéalement, après plusieurs interventions, LGPS devrait être capable de vider le bac de manière autonome, mais nous n'avons pas pu le tester.

En conclusion, cette thèse a permis de poser les bases pour des approches data-efficaces et adaptées à des tâches spécifiques en matière de préhension robotique. Les méthodes développées ont montré leur potentiel pour améliorer les performances de la préhension robotique dans divers scénarios et environnements. Toutefois, il subsiste de nombreux défis et opportunités pour optimiser et étendre ces approches, afin de permettre aux robots de maîtriser des compétences de préhension et de manipulation toujours plus complexes et adaptées à un large éventail de situations.

Annexe A

Intégration des éléments matériels et logiciels

Dans ce chapitre, nous allons décrire les composants matériels et logiciels utilisés dans notre thèse. En termes de matériel, nous avons utilisé un bras robotique Franka Emika Panda (Sec. A.1) équipé de son préhenseur d'origine, auquel nous avons ajouté une caméra RGB-D Realsense D415 (Sec. A.2) et des capteurs tactiles Digit (Sec. A.3). Des pièces imprimées en 3D ont été utilisées pour fixer les capteurs et les câbles au robot. Nous avons utilisé une architecture à deux ordinateurs (Fig. A.1) : l'un équipé d'un noyau en temps réel pour commander le robot et l'autre équipé d'une carte graphique pour effectuer les calculs nécessaires à la génération des préhensions.

Les composants logiciels que nous avons utilisés répondent aux besoins suivants :

— la récupération de démonstrations de saisie de la part des utilisateurs (Sec. A.5 et Sec. A.1.2) ;
— la sauvegarde et l'organisation des données provenant du robot, des algorithmes, des capteurs et des experts (Sec. A.4) ;
— la génération de préhensions (Sec. 6.1) ;
— le contrôle du robot pour effectuer les actions de préhension (Sec. A.1) ;
— l'évaluation de nos algorithmes et des autres méthodes de l'état de l'art en ligne, en simulation et hors ligne (bases de données préexistantes).

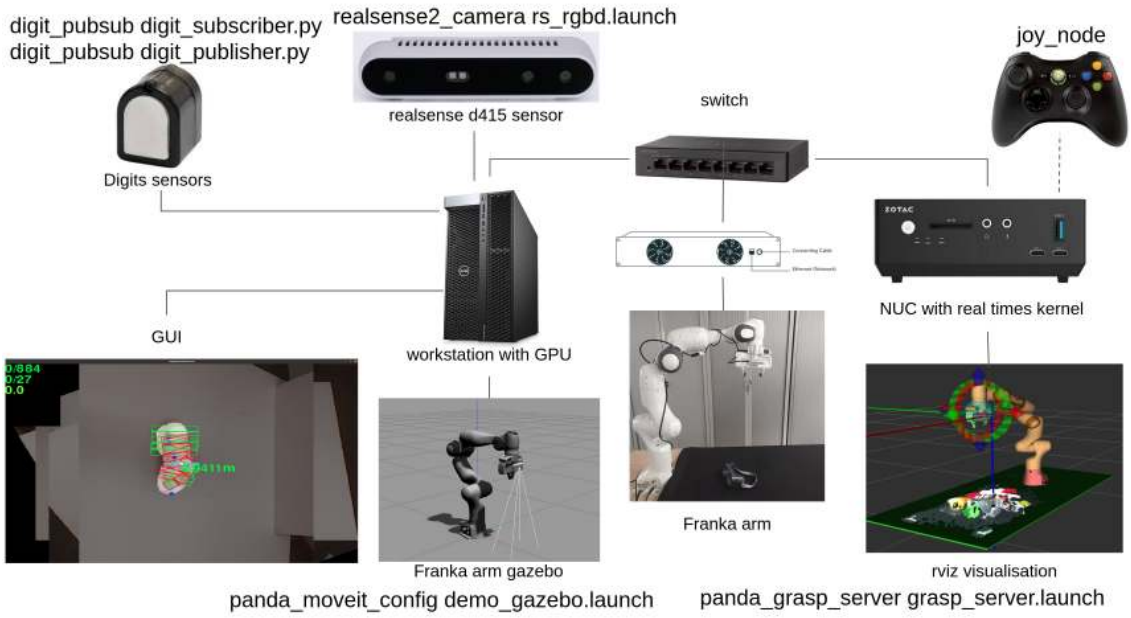


FIGURE A.1 – Grâce à la modularité de ROS, chaque composant peut s'interfacer avec l'interface graphique (le robot ne peut pas être simultanément contrôlé et simulé).

Robot Operating System (ROS) fournit un ensemble d'outils et de bibliothèques pour le développement d'applications robotiques. Il est utile car il permet aux développeurs de créer des applications robotiques plus rapidement et plus facilement en leur fournissant un ensemble de services et d'interfaces de programmation (API) communs qui peuvent être utilisés pour contrôler une grande variété de robots différents. Les nœuds peuvent communiquer entre eux par le biais de requêtes et de réponses, les publishers sont des composant qui envoient des messages sur un topic spécifique, et les subscribers reçoivent les messages d'un topic. Cela permet à différents composants de communiquer entre eux pour émettre et recevoir des données ou exécuter des programmes.

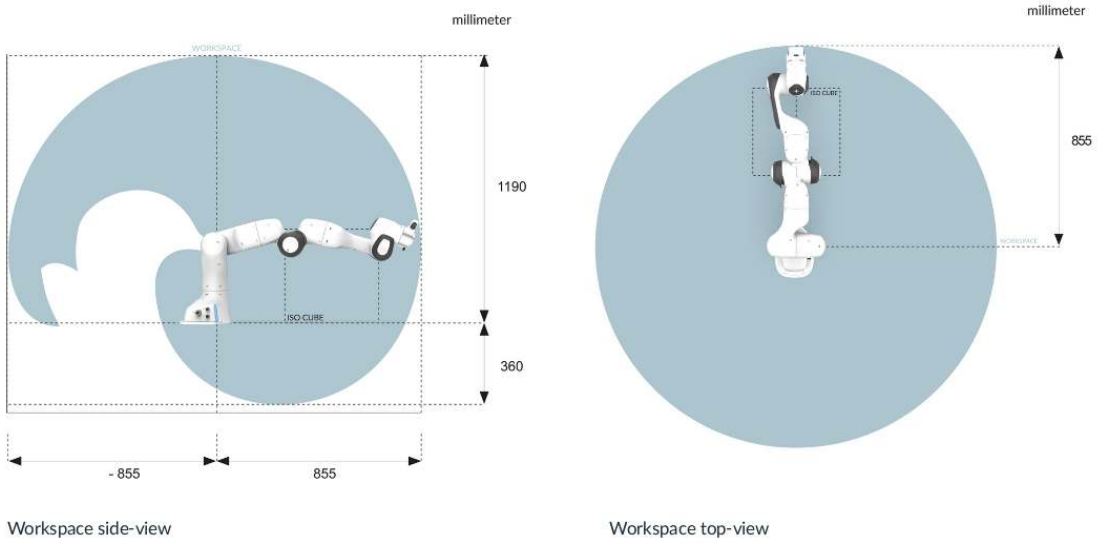
L'un des principaux avantages de ROS est qu'il permet aux développeurs de créer un code modulaire et réutilisable qui peut être facilement intégré à d'autres composants d'un système robotique. Ce qui permet donc de faire communiquer différents programmes sur plusieurs machines et d'utiliser ou de modifier des modules ROS existants comme le contrôle de la caméra ou la planification de mouvements du et leur exécution.

Cette approche modulaire permet également de ne faire fonctionner qu'une partie des modules développés en fonction des besoins, ainsi l'interface graphique détaillée en (Sec. A.5) peut fonctionner seule (évaluation/étiquetage hors ligne), en simulation (gazebo) ou avec le vrai robot, en fonction des valeurs rentrées dans différents fichiers de configurations.

Les variables d'environnement *ROS_MASTER_URI* et *ROS_IP* sont utilisées pour spécifier l'URI (Uniform Resource Identifier) du nœud maître ROS et l'adresse IP (Internet Protocol) des machines. Le nœud maître ROS est le point central de communication dans le réseau ROS. Il tient à jour un registre de tous les autres nœuds du système et est chargé de coordonner leurs activités.

A.1 Bras robotique Franka

Le robot Franka est un bras articulé possédant 7 degrés de liberté et il est équipé de capteurs de couple au niveau de ses axes. Équipé du préhenseur Franka hand, il est capable d'exercer une force de préhension de 70 N en continu et de 140 N au maximum, à une vitesse de fermeture de 50 mm/s par doigt. Bien qu'il ne soit pas équipé de capteurs de couple ou de force, nous avons ajouté des capteurs tactiles Digit (Sec. A.3). Cela lui permet de soulever jusqu'à 3 kg avec une ouverture maximale de sa pince de 80 mm.



.

FIGURE A.2 – Espace de travail du robot.

L'espace de travail du robot est illustré dans la figure A.2. Il peut partager un environnement avec un opérateur humain grâce à sa capacité de détection des collisions. En effet, il peut détecter une collision en un temps allant de 2 ms à 100 ms. Lorsqu'une collision est détectée, le robot arrête son mouvement.

Le robot propose également un mode collaboratif où il suit l'enseignement de l'opérateur humain. Il est possible d'accéder à ce mode en appuyant simplement sur un bouton situé près de l'effecteur du robot. À chaque pression sur ce bouton, le robot compense la gravité et la friction pour réduire la perception de la gravité, ce qui permet à l'utilisateur de le guider manuellement.

Pour régler les paramètres de son contrôle (c'est-à-dire l'impédance, le temps de réaction et l'accélération maximale), nous pouvons accéder à l'interface du robot lors de sa phase de démarrage.

A.1.1 Commandes de haut niveau

Pour contrôler les mouvements du robot, nous avons utilisé MoveIt [10], une bibliothèque de planification de mouvement qui communique avec le robot via ROS. MoveIt possède

10. `https://moveit.ros.org/`

une interface en python et en C++, et nous avons utilisé python pour nos expériences.

Nous avons modifié les fichiers URDF [11] (Unified Robot Description Format), un format XML pour représenter un modèle de robot, chargés par MoveIt (Fig. A.8) pour nos besoins :

— ajout d'un lien virtuel comme point central de l'outil du robot (TCP), ce qui permet une meilleure planification des saisies ;

— ajout du modèle de la caméra montée sur l'outil terminal du robot avec l'option de compatibilité [12] avec Gazebo [13] (un simulateur de robotique 3D open-source).

A.1.2 Pilotage par joystick

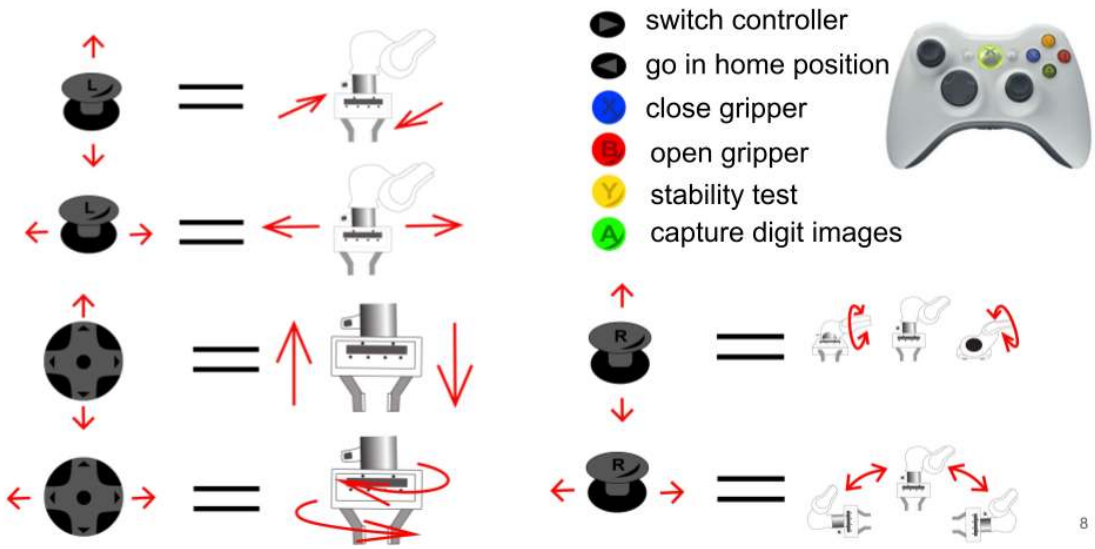


FIGURE A.3 – Le stick analogique gauche du joystick permet de gérer les déplacements en x-y, le D-pad permet de se déplacer en z et de contrôler le roulis de l'axe de l'effecteur, et le stick analogique droit gère les deux autres axes de rotation. Les boutons X et B contrôlent l'ouverture et la fermeture du préhenseur, la touche A enregistre les images du capteur Digit et Y permet de réaliser une trajectoire d'excitation constituée de roto-translations rapides, en particulier de rotations autour de l'axe de l'effecteur.

Afin de collecter certaines démonstrations, un contrôleur de vitesse cartésien a été utilisé pour piloter le mouvement du bras robotique. Le bras est commandé en spécifiant les vitesses souhaitées de l'effecteur dans les directions x, y et z, ainsi que les vitesses de rotation souhaitées autour de chacun de ces 3 axes.

Le contrôleur utilise les entrées de vitesse de la manette Xbox 360 pour calculer les vitesses des articulations nécessaires pour déplacer l'effecteur final aux vitesses désirées.

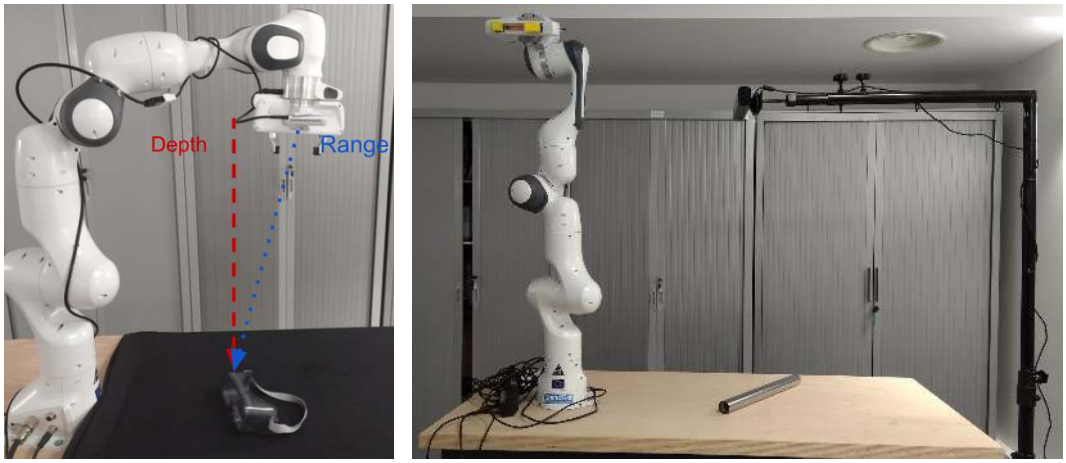
11. `https://gitlab.inria.fr/CHIST-ERA-Heap/public/franka_ros`
12. `https://github.com/pal-robotics/realsense_gazebo_plugin`
13. `https://gazebosim.org/home`

Ces vitesses d'articulation sont ensuite envoyées aux moteurs du bras, qui actionnent les articulations pour produire le mouvement souhaité. Cela permet notamment de contrôler l'approche du bras dans une seule direction (Fig. A.3), ce que nous avons utilisé pour collecter et tester des exemples de saisies en profondeur tout en maintenant les mêmes positions x, y et orientations (Chap. 4). Les boutons de la manette sont également utiles pour effectuer rapidement des tests et déboguer.

A.2 Caméra Realsense D415

(a) image RGB (b) image de profondeur

FIGURE A.4 – Exemple de scène capturée par la caméra Realsense D415, l'image de profondeur à droite est générée en associant une valeur entre 0 et 255 pour chaque pixel de l'image ; Les pixels les plus éloignés de la caméra apparaissent plus clairs.



(a) Caméra montée sur le robot (b) Caméra fixe

FIGURE A.5 – À gauche, illustration de la profondeur retournée par la caméra avec la caméra montée sur le robot. À droite, la caméra est montée à l'extérieur du robot et observe son espace de travail.

Nous avons choisi la caméra RealSense D415 d'Intel pour nos expériences, après avoir considéré d'autres options, détaillées dans [62]. Elle est alimentée en USB 3 [14] et équipée d'une paire de caméras infrarouges, d'une caméra RGB et d'un projecteur infrarouge permettant de capturer une image en couleur et un nuage de points de leur profondeur (Fig. A.4) [15]. Le calcul de la profondeur est réalisé par stéréo-vision, via les deux caméras infrarouges parallèles, les calculs étant effectués par le processeur de la caméra. Le projecteur infrarouge projette un motif statique afin d'améliorer la précision du calcul de la profondeur. Les valeurs de profondeur des pixels sont les mesures par rapport aux plans parallèles des caméras (Depth) et non la distance absolue (Range) (voir Fig. A.5 (a)).

A.2.1 Calibrage de la caméra

L'objectif de cette étape est de réaliser la calibration des paramètres extrinsèques et intrinsèques de la caméra. Les paramètres extrinsèques correspondent à la pose de la caméra par rapport à l'outil terminal du robot si la caméra est montée sur le bras du robot, ou à sa pose par rapport au référentiel mondial si la caméra est montée à l'extérieur des liens du bras du robot, de sorte que sa pose ne change pas lorsque le bras du robot se déplace (Fig. A.5).

Le modèle du sténopé (Fig. A.5) modélise les relations de passage du repère monde au repère caméra. Il permet d'exprimer la projection du repère caméra dans le plan image et d'appliquer la transformation affine qui conduit aux coordonnées de l'image A.1. Les points du monde (X, Y, Z) sont transformés en coordonnées de la caméra en utilisant les paramètres extrinsèques. Les coordonnées de la caméra sont ensuite transposées dans le plan de l'image à l'aide des paramètres intrinsèques.

$$s * \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} & R_{3\times3} & & \begin{matrix} t_x \\ t_y \\ t_z \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{A.1}$$

La calibration intrinsèque permet de déterminer les paramètres suivants :

— c_x, c_y : Centre optique (le point principal), en pixels
— f_x, f_y : Longueur focale, en pixels
— s : Coefficient d'obliquité, qui est non nul si les axes de l'image ne sont pas perpendiculaires

Cette calibration est réalisée en utilisant le logiciel Realsenseviewer [16].

La calibration extrinsèque permet de déterminer les éléments suivants :

— $R_{3\times3}$: Matrice de rotation permettant de passer du repère lié à l'espace de travail au repère lié à la caméra
— t_x, t_y, t_z : Composantes du vecteur de translation permettant de passer du repère lié à l'espace de travail au repère lié à la caméra

14. Le fichier des pièces consolidant le branchement est téléchargeable au lien suivant `https://mybox.inria.fr/d/6f158cdb2742419a9706/`

15. La pièce imprimable en 3D permettant de fixer la caméra à l'outil terminal du Franka peut être trouvée sur `https://franka.world/resources`

16. `https://github.com/IntelRealSense/librealsense`

A.2.1.1 Cas de la caméra fixe - Perspective-n-Point

Le problème de Perspective-n-Point (PnP) est une question de vision par ordinateur qui consiste à estimer la position et l'orientation d'un objet connu à partir d'une image. On peut donc l'appliquer pour trouver la transformation ${}^{C}\mathcal{T}_{R}$ entre le repère de la caméra et celui de la base du robot. On l'appelle Perspective-n-Point parce qu'il utilise n points sur l'objet pour déterminer sa pose en résolvant un système d'équations pour déterminer la position et l'orientation de l'objet dans l'espace 3D, en se basant sur les coordonnées des n points de l'image.

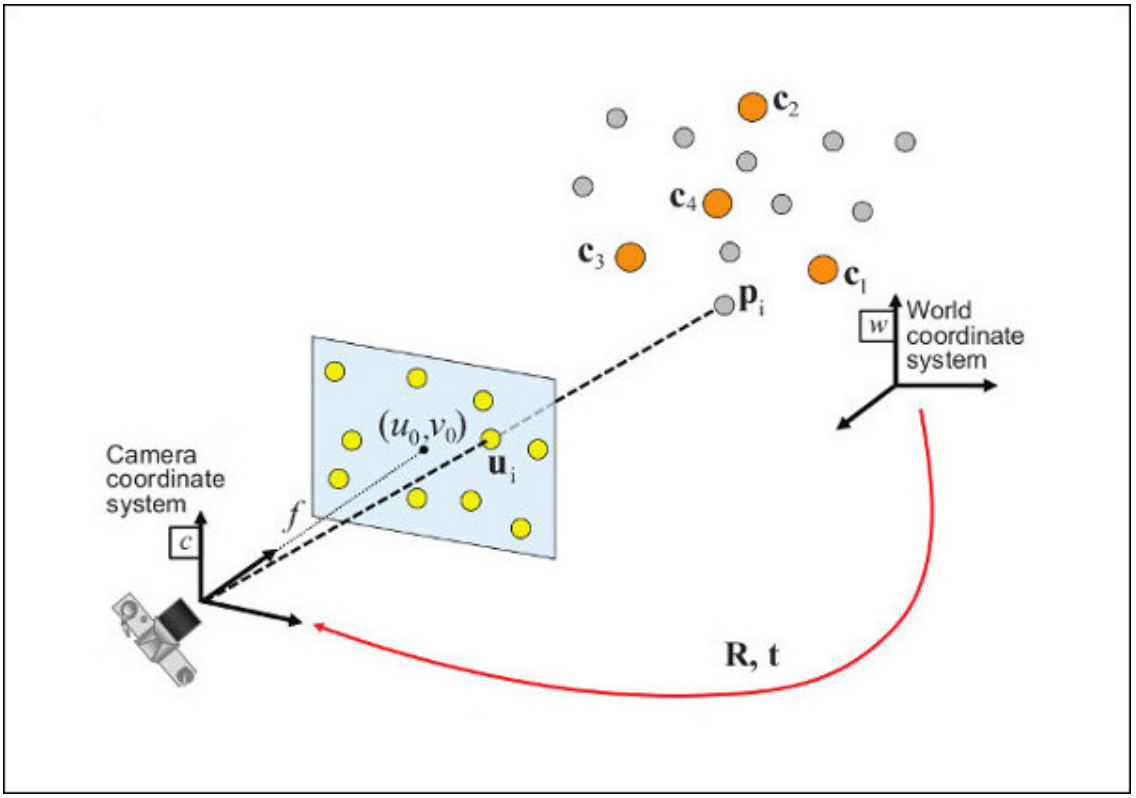


FIGURE A.6 – Image provenant de `https://docs.opencv.org/4.x/d5/d1f/calib3d_solvePnP.html`

Une fois qu'un objet est détecté dans le repère de la caméra (coordonnées en pixels), on peut obtenir sa position dans l'espace en utilisant les paramètres intrinsèques et sa profondeur z dans le nuage de points (X_c, Y_c, Z_c). En plaçant l'objet dans la pince du robot (Fig. A.7), on peut obtenir sa pose dans le repère de la base du robot en utilisant l'interface de commande MoveIT (cinématique directe) (X_r, Y_r, Z_r).

$$\begin{pmatrix} X_c \\ X_y \\ Z_c \\ 1 \end{pmatrix} = {}^{C}\mathcal{T}_{R} * \begin{pmatrix} X_R \\ Y_R \\ Z_R \\ 1 \end{pmatrix} \tag{A.2}$$

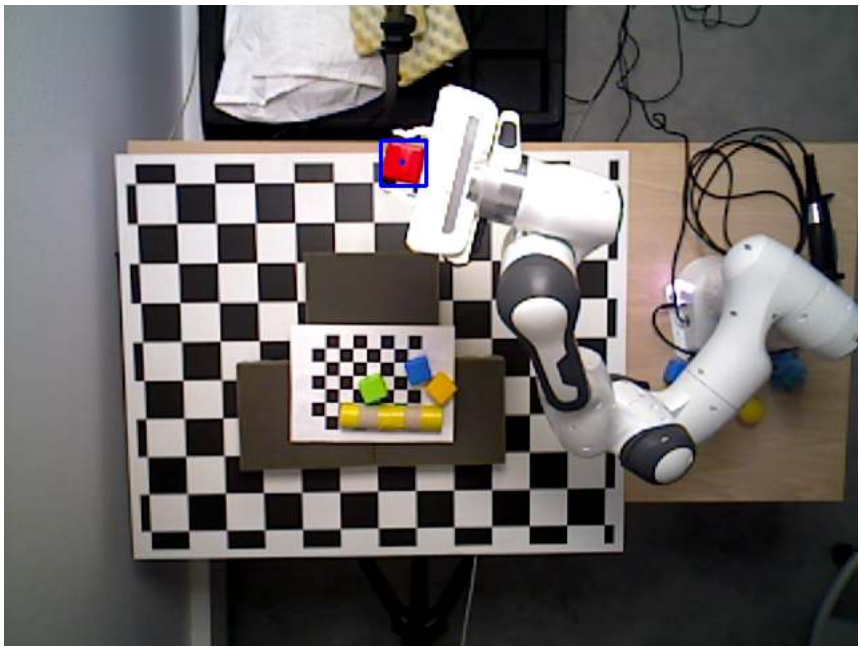


FIGURE A.7 – Identification de la position de l'effecteur final en utilisant le cube rouge comme marqueur.

Après avoir collecté un ensemble de points dans les deux référentiels, nous pouvons calculer la matrice de transformation entre eux en utilisant les solveurs d'OpenCV [17].

A.2.1.2 Cas de la caméra montée sur l'outil terminal

Lorsque la caméra est montée sur l'outil terminal du robot, on peut également trouver la transformation entre le repère de la caméra et celui de la base du robot en utilisant la géométrie connue des modèles de la caméra [18], de la pièce reliant la caméra à l'outil terminal, et du robot [19].

17. https://github.com/heap-chist-era/Test_Dexnet/blob/master/main_calibration.py
18. https://github.com/IntelRealSense/realsense-ros/tree/development/realsense2_description/urdf
19. https://github.com/frankaemika/franka_ros/tree/develop/franka_description/robots/common

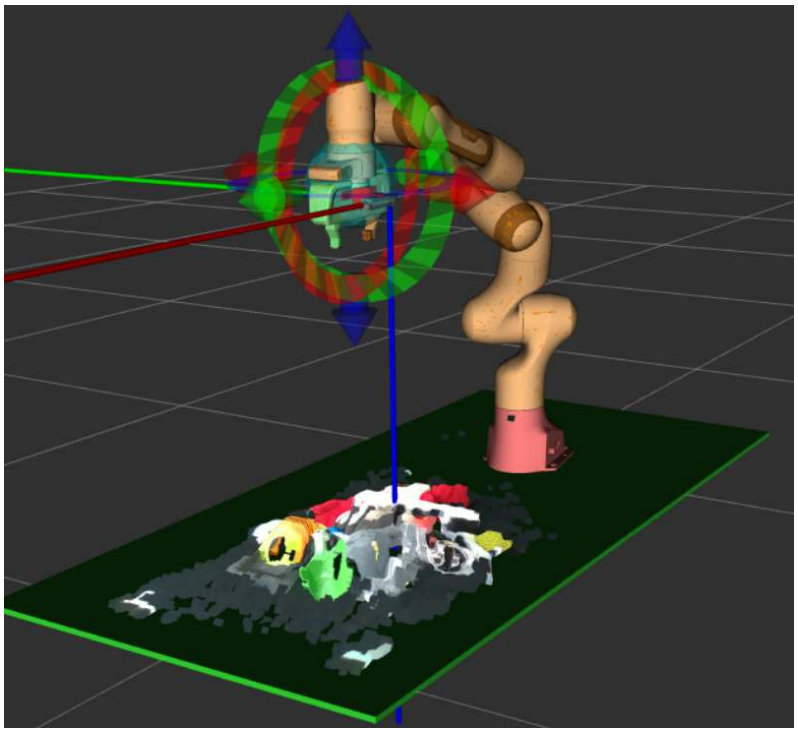


FIGURE A.8 – Visualisation du robot équipé du modèle de la caméra et de sa fixation, ainsi que le nuage de points issu de la caméra dans rviz. Le modèle vert correspond à la table sur laquelle les objets reposent ; la planification des trajectoires prend en compte cette table pour éviter les collisions.

A.3 Capteur Digit

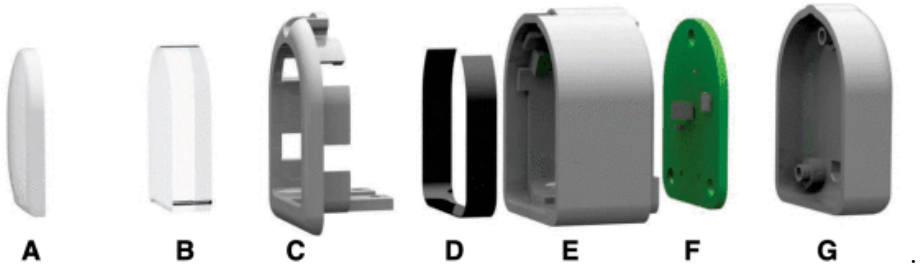
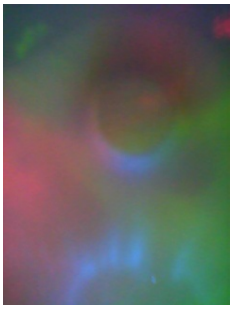


FIGURE A.9 – Vue éclatée d'un capteur Digit [92]. A) élastomère, B) fenêtre acrylique, C) support à encliquetage, D) circuit imprimé d'éclairage, E) boîtier plastique, F) circuit imprimé de la caméra, G) boîtier arrière.

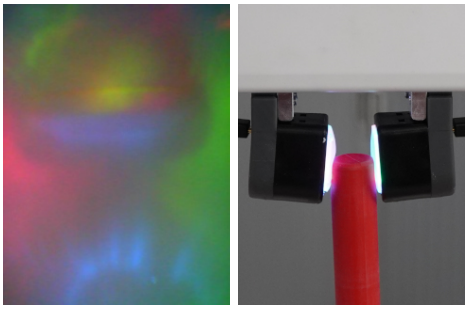
Le capteur tactile basé sur la vision Digit [92] est composé d'une fenêtre en acrylique située sur la face intérieure de la structure du capteur, éclairée par des LED de trois couleurs (rouge, vert et bleu), et sur la face extérieure recouverte d'un élastomère (dispositif permettant de mesurer l'élasticité d'un matériau). Il est également équipé d'une caméra CMOS qui capture l'image de la fenêtre acrylique, utilisée comme réponse du capteur

(Fig. A.9). Grâce à un support imprimé en 3D [20], les capteurs sont montés sur l'effecteur du bras robotique Franka et utilisés comme "doigts" dans les tâches de préhension, comme le montre l'image (Fig. A.10).

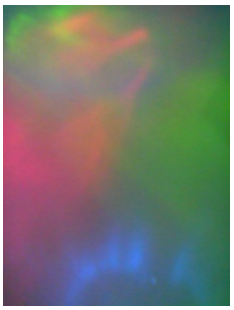
Lorsqu'un objet entre en contact ou exerce une pression sur la surface du capteur, la fenêtre acrylique se déforme et l'effet de l'objet est visible sur l'image de sortie. Le Digit se connecte à un ordinateur via un port USB, et sa commande peut être effectuée soit par l'interface du fabricant, soit par la bibliothèque OpenCV. Des exemples de la réponse du capteur pour différents objets sont présentés dans (Fig. A.10). Les objets à surface plane sont les moins bien détectés par le Digit, contrairement aux petits objets ou aux objets avec des bords. Les objets courbes, lorsqu'ils sont saisis en un point autre que le centre, sont également moins perceptibles.



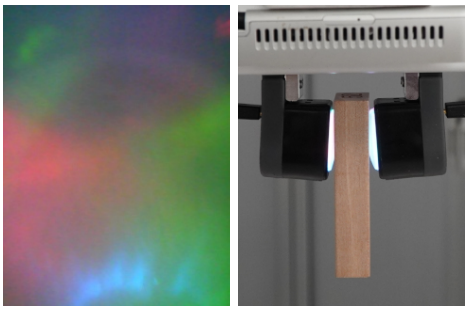
(a) Petit objet serré par les deux capteurs



(b) Gros objet dépassant le capteur



(c) Objet incurvé



(d) Objet à surface plane en contact avec toute la surface du capteur

FIGURE A.10 – La réponse du Digit à différents types de contact.

L'analyse des images capturées par le capteur Digit peut être réalisée en utilisant des algorithmes de traitement d'image et d'apprentissage automatique. Ces techniques permettent de caractériser les objets saisis et d'optimiser la préhension en fonction des propriétés de l'objet et des contraintes du bras robotique. L'intégration du capteur Digit avec des modèles d'apprentissage automatique peut offrir une meilleure compréhension de la manipulation des objets et améliorer la performance des systèmes robotiques dans des tâches de préhension complexes.

20. le fichier du support est téléchargeable au lien suivant `https://mybox.inria.fr/d/79f2ff9cbe984617afed/`

A.4 Base de données

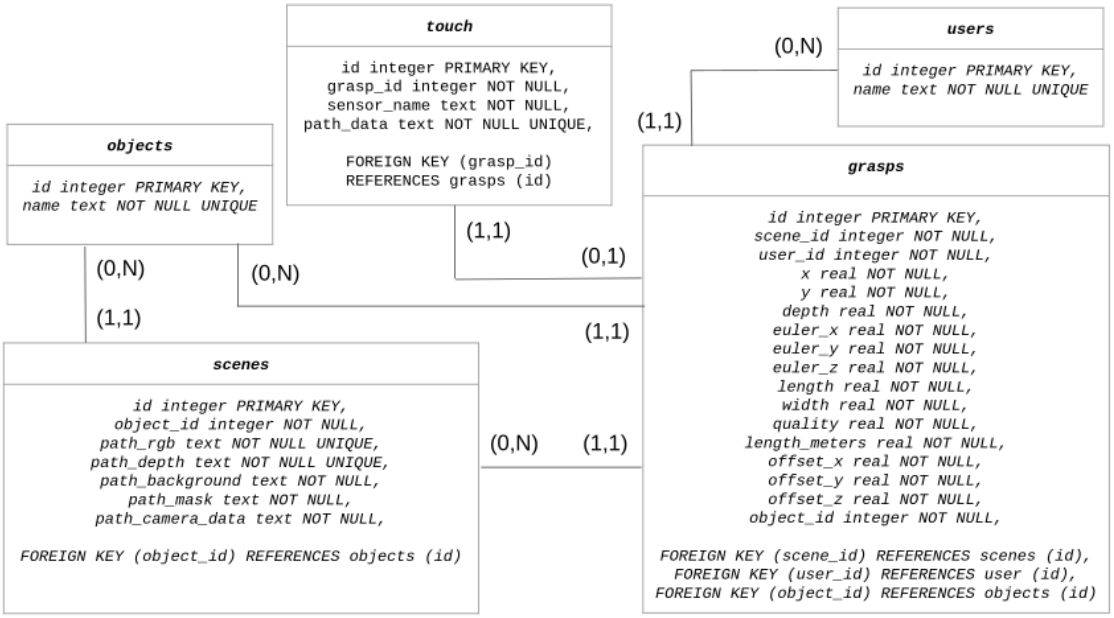


FIGURE A.11 – La base de données relationnelle composée de 5 tables (objects, scenes, grasps, touch, users).

Lorsque la caméra capture une scène RGB-D contenant des objets à saisir, celle-ci peut être ajoutée à la base de données, et un objet lui est associé (un objet 'tas' ou 'inconnu' peut être utilisé pour passer cette étape). Les données RGB-D et les paramètres intrinsèques et extrinsèques de la caméra sont sauvegardés, une image RGB de l'arrière-plan et un masque de segmentation peuvent être spécifiés.

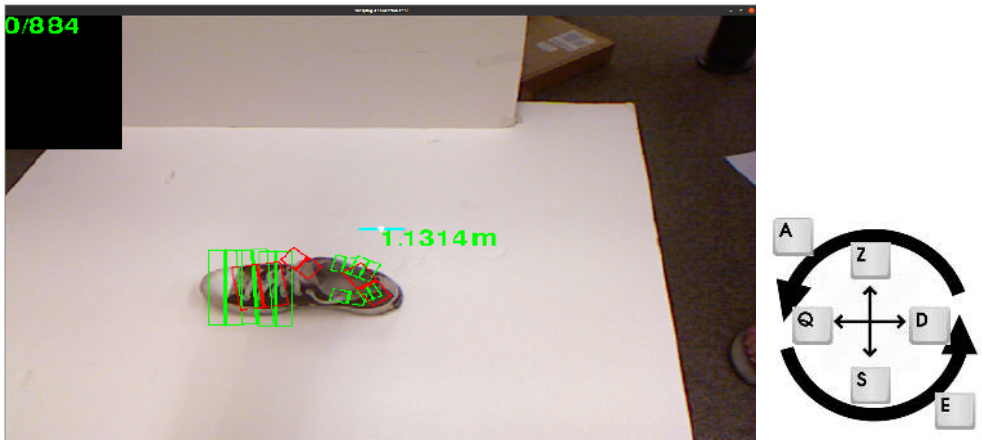
Chaque scène peut avoir des préhensions associées à elle-même, ces saisies appartenant à un utilisateur. Ces utilisateurs peuvent être les utilisateurs de l'interface graphique (Sec. A.5) ou des générateurs de préhensions (section 6.1). Si l'objet de la scène n'est pas un 'tas', l'objet saisi est le même que celui associé à la scène. Dans le cas contraire, l'identifiant de l'objet saisi est associé. Certaines préhensions peuvent également avoir des données provenant du capteur Digit.

La mise en œuvre d'une base de données SQL permet un moyen structuré de stocker et d'organiser les données tout en assurant la validité des données stockées. De plus, le langage SQL permet de créer des requêtes complexes capables de récupérer et de traiter les données. Cela est particulièrement utile pour les tester (par exemple, sélectionner toutes les scènes d'un objet en particulier, ou les scènes ne possédant pas de démonstrations de préhensions fournies par l'utilisateur lors de la phase d'étiquetage).

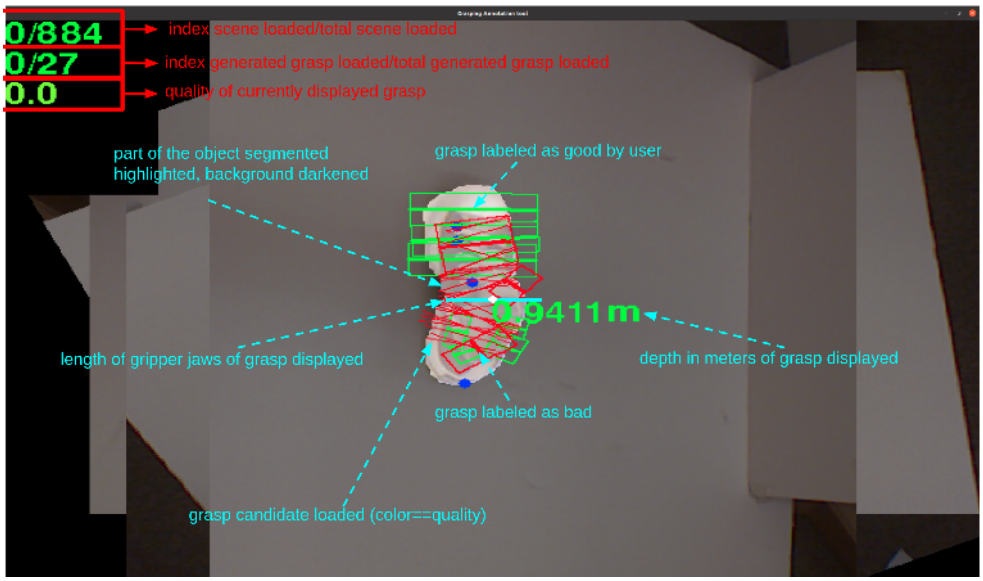
Cette approche structurée permet d'améliorer l'efficacité de l'analyse des données et facilite l'intégration de nouveaux algorithmes d'apprentissage automatique pour l'optimisation des préhensions. En outre, l'utilisation d'une base de données relationnelle facilite la collaboration entre les chercheurs et les ingénieurs travaillant sur le projet, en offrant un moyen standardisé de partager et de gérer les données.

A.5 Interface graphique

L'interface graphique utilisateur (GUI) [21] se compose d'éléments visuels et de fonctionnalités utiles pour étiqueter les données nécessaires pour l'apprentissage. Elle permet également aux utilisateurs d'interagir avec le robot de manière simple et intuitive, ce dont nous avions besoin pour réaliser l'expérience détaillée au chapitre 5.



(a) Scène chargée et touches du clavier permettant de manuellement fournir une saisie.



(b) Scène après la génération de saisie.

FIGURE A.12 – (a) Une scène chargée dans l'interface graphique, (b) quelques exemples de touches du clavier permettant à l'utilisateur d'interagir avec le logiciel en se déplaçant dans l'image (c) descriptions de ce qui est affiché à l'écran après avoir généré des préhensions.

Des classes Python 3 ont été développées pour offrir une interface de programmation pour gérer les capteurs, le robot, ainsi que la génération et l'évaluation des préhensions.

21. `https://gitlab.inria.fr/CHIST-ERA-Heap/public/latent_space_gp_selector`

Les classes proposent des méthodes (c'est-à-dire les fonctions définies dans la classe) et des attributs (c'est-à-dire les variables associées à la classe ou à ses objets) et sont instanciées en fonction d'un fichier de configuration fourni en paramètre :

- camera : souscrit aux publishers/services liés à la caméra Realsense D415 [22], récupère à un instant donné les données RGB-D et les paramètres intrinsèques et extrinsèques de la caméra ;
- camera digit : souscrit aux publishers/services liés aux capteurs Digit [23], récupère les images des capteurs et réalise les calculs des ellipses de contacts et la prédiction de la présence d'un objet entre les capteurs ;
- grasps evaluator : réalise les prédictions de qualité, ouverture du préhenseur et profondeur de la saisie ;
- panda control : souscrit aux publishers/services liés au robot [24] et envoie des commandes de haut niveau pour contrôler le robot ;
- database : gère la sauvegarde des données sur le disque et dans la base de données [25] (section 6.1) ;
- segmentation : réalise la segmentation d'une scène RGB-D (Sec. A.6) ;
- grasps generators : souscrit aux publishers/services liés à la génération de préhensions [26] ; (section 6.1).
- cvgrasp : réalise une génération de candidats basés sur des algorithmes de vision par ordinateur (Sec. 6.1.4).

Ces classes sont instanciées par le logiciel proposant l'interface graphique utilisateur ; l'affichage peut également être désactivé pour seulement servir d'interface utile pour écrire les scripts pour nos expériences (entraîner nos modèles, évaluer hors ligne les performances via le IoU test).

Dans notre cas, l'interface (Fig. A.12) répond au besoin suivant :

- Fournir un retour visuel à l'utilisateur, venant de la caméra du robot ou de données capturées au préalable, ce qui est utile pour déboguer et comprendre le comportement du robot ;
- Collecter les préhensions labellisées par un utilisateur ;
- Offrir une interface de programmation afin d'accélérer l'écriture de nos expériences.

A.6 Segmentation des scènes RGB-D

La segmentation est le processus qui consiste à diviser une image en différentes régions ou "segments", chaque "segment représentant un objet ou une partie d'objet différent dans l'image. Ce procédé est utile pour une tâche de préhension robotique basée sur la vision car il permet au robot d'identifier et d'isoler l'objet qu'il doit saisir. En segmentant l'image, le robot peut concentrer son attention sur l'objet spécifique qu'il doit saisir, et ignorer

22. https://github.com/IntelRealSense/realsense-ros
23. https://gitlab.inria.fr/CHIST-ERA-Heap/digit_franka/
24. https://gitlab.inria.fr/CHIST-ERA-Heap/public/panda_ros_common/,https://gitlab.inria.fr/CHIST-ERA-Heap/public/panda_grasp_server/,https://gitlab.inria.fr/CHIST-ERA-Heap/public/franka_ros/,https://gitlab.inria.fr/CHIST-ERA-Heap/public/franka_ros_gazebo/,https://gitlab.inria.fr/CHIST-ERA-Heap/public/panda_moveit_config/,https://gitlab.inria.fr/CHIST-ERA-Heap/public/panda_moveit_config_gazebo/
25. https://gitlab.inria.fr/CHIST-ERA-Heap/public/heap_human_preference_dataset/
26. https://gitlab.inria.fr/CHIST-ERA-Heap/public/grasping-benchmarks-panda/

les autres objets qui peuvent être présents dans la scène. Cela peut aider le robot à être plus précis et efficace dans sa tâche de saisie. Cette étape est optionnelle, mais la plupart des algorithmes de synthèse saisies peuvent soit prendre en entrée le masque de la segmentation ou peuvent directement segmenter les données RGB-D.

Il existe plusieurs façons d'effectuer une segmentation en vision par ordinateur, et les approches les plus courantes impliquent l'utilisation d'algorithmes d'apprentissage automatique supervisés ou non supervisés. Les méthodes supervisées nécessitent une base de données d'images étiquetées, où chaque pixel de l'image se voit attribuer une étiquette de classe (telle qu'objet" ou "arrière-plan"). L'algorithme est entraîné sur cette base de données, puis il peut être appliqué à de nouvelles images pour identifier et segmenter les objets de la scène. Les méthodes non supervisées, quant à elles, ne nécessitent pas de données étiquetées et s'appuient plutôt sur des algorithmes de regroupement des pixels de l'image en différents segments.

Récemment, les approches basées sur l'apprentissage profond ont gagné en popularité pour la segmentation des images, notamment les réseaux de neurones convolutionnels (CNN) [102] et les réseaux de neurones entièrement convolutionnels (FCN) [6]. En plus des CNN et FCN mentionnés précédemment, d'autres architectures de réseaux de neurones populaires pour la segmentation d'images incluent U-Net [150] et DeepLabv3 [34]. U-Net est un réseau de neurones convolutionnel spécialement conçu pour la segmentation biomédicale d'image, avec une architecture en forme de U qui permet une résolution d'image élevée et une localisation précise des objets. DeepLabv3, quant à lui, est une architecture de segmentation d'image qui utilise des convolutions dilatées pour capturer efficacement plusieurs échelles d'information et améliorer la performance de segmentation. Les méthodes de segmentation basées sur l'apprentissage profond ont généralement de meilleures performances que les méthodes classiques, mais elles nécessitent des ensembles de données étiquetées volumineux et beaucoup de puissance de calcul pour l'entraînement.

Parmi les techniques de segmentation non supervisées, on peut citer les algorithmes de regroupement tels que les k-means et les modèles de Modèle de Mélange Gaussien (GMM) [212, 213]. Ces algorithmes regroupent les pixels de l'image en fonction de leur couleur ou de leur intensité, et chaque groupe est traité comme un segment distinct. Un autre exemple de technique non supervisée est la segmentation basée sur les graphes [151, 167, 104], qui traite l'image comme un graphe et applique des algorithmes de théorie des graphes pour identifier les segments dans l'image. Nous avons implémenté les méthodes suivantes :

— **crop** : On met à l'arrière-plan les pixels sur les côtés de l'image en fonction des plages de valeurs spécifié. Cela permet de facilement retirer les bords en dehors de la table ou les zones vu par la caméra en dehors de l'espace de travail du Robot.
— **closed edge** : **1-** Utiliser le détecteur de bords de Canny pour détecter les bords de l'image. Le détecteur de contours de Canny [30] est un algorithme de vision par ordinateur qui utilise les informations de gradient pour identifier les pixels qui appartiennent aux contours d'une image. **2-** Créer un masque binaire en appliquant un seuil à la carte des contours produite par le détecteur de contours de Canny. Ce masque aura des pixels avec une valeur de 1 aux endroits où les bords ont été détectés et des pixels avec une valeur de 0 partout ailleurs. **3-** Utiliser des opérations morphologiques, telles que la dilatation et l'érosion, pour nettoyer le masque binaire et combler les lacunes ou les trous dans les bords détectés. **4-**

Utilisez le masque binaire rempli comme masque de segmentation pour séparer les objets de premier plan de l'arrière-plan dans l'image originale.

— **saliency** : Dans la segmentation par saillance [183], l'algorithme calcule d'abord une carte 2D de l'image où chaque pixel se voit attribuer une valeur indiquant son degré de saillance (La saillance désigne la propriété d'une image qui fait ressortir certaines régions ou attire l'attention de l'observateur.). La carte de saillance est calculée en utilisant diverses caractéristiques visuelles, telles que la couleur, l'intensité, la texture et l'emplacement spatial, ainsi que des modèles statistiques de l'attention visuelle humaine.

— **HSV** : HSV est un espace couleur qui représente les couleurs à l'aide de trois canaux : la teinte, la saturation et la valeur. La teinte représente l'information sur la couleur, la saturation représente l'intensité ou la pureté de la couleur, et la valeur représente la luminosité de la couleur. La segmentation se fait selon 3 étapes : **1-** Convertir l'image de l'espace couleur RGB à l'espace couleur HSV. **2-** Définir une plage de valeurs de teinte, de saturation et de valeur appartenant à l'arrière plan et segmenter l'image. **3-** Utiliser des opérations morphologiques, telles que la dilatation et l'érosion, pour nettoyer le masque binaire.

— **background** : Gaussian Mixture-based Background/Foreground Segmentation Algorithm [212, 213] repose sur l'hypothèse que les valeurs d'intensité de chaque pixel de l'image peuvent être modélisées à l'aide d'un modèle de mélange de gaussiennes. Ce GMM est ensuite utilisé pour classer chaque pixel de l'image comme appartenant à l'arrière-plan ou au premier plan en fonction de la probabilité qu'il provienne du modèle d'arrière-plan.

— **depth** : Les données de profondeurs de la caméra RGB-D peuvent être utilisées pour segmenter les objets de la table en ne gardant que les points dont la profondeur est suffisamment éloignée de celle de la table.

— **ransac** : RANSAC (Random Sample Consensus) utilise les données de profondeurs de la caméra RGB-D. Il peut être utilisé pour segmenter les objets reposant sur la table en adaptant un modèle de la table à un ensemble de points de données et en ne gardant que les points qui ne correspondent pas au modèle. RANSAC sélectionne itérativement un sous-ensemble aléatoire des données d'origine. Ces données sont d'hypothétiques données pertinentes et cette hypothèse est ensuite testée comme suit : **1-** Un modèle est ajusté aux données pertinentes hypothétiques, c'est-à-dire que tous les paramètres libres du modèle sont estimés à partir de ce sous-ensemble de données. **2-** Toutes les autres données sont ensuite testées sur le modèle précédemment estimé. Si un point correspond bien au modèle estimé, alors il est considéré comme une donnée pertinente candidate. **3-** Le modèle estimé est considéré comme correct si suffisamment de points ont été classés comme données pertinentes candidates. **4-** Le modèle est ré-estimé à partir de ce sous-ensemble des données pertinentes candidates. **5-** Finalement, le modèle est évalué par une estimation de l'erreur des données pertinentes par rapport au modèle. Cette procédure est répétée un nombre fixe de fois, chaque fois produisant soit un modèle qui est rejeté parce que trop peu de points sont classés comme données pertinentes, soit un modèle réajusté et une mesure d'erreur correspondante.

— **grabcut** : GrabCut est une technique semi-supervisée, ce qui signifie qu'elle utilise des données étiquetées et non étiquetées pour segmenter l'image. Des pixels

appartenant à l'avant-plan et arrière-plans sont fournis et des modèles statistiques sont utilisés pour segmenter l'image en régions de premier plan et d'arrière-plan. En utilisant une segmentation préliminaire avec les algorithmes décrits ci-dessus, l'algorithme GrabCut peut raffiner le masque, améliorant la segmentation mais augmentant le temps pour la réaliser.

Une première estimation de paramètres adéquats est obtenue en utilisant une interface graphique avec curseurs, comme le montre la Fig. A.13. Cette interface permet de visualiser en temps réel les effets des modifications apportées aux paramètres de segmentation, facilitant ainsi le choix des valeurs optimales pour chaque méthode.

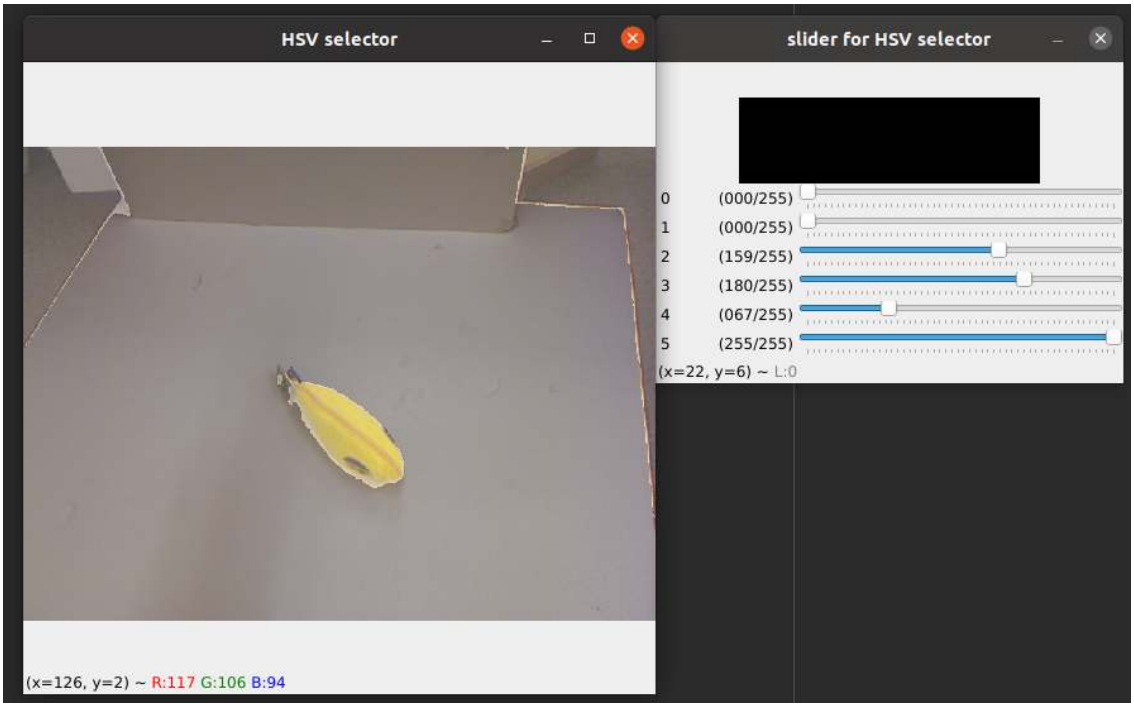


FIGURE A.13 – A gauche, la fenêtre affiche l'image segmentée (les parties sombres correspondent à l'arrière-plan, les parties en surbrillance aux objets). A droite les curseurs correspondent aux paramètres de la méthodes (ici, il y a 6 curseurs correspondant à la plage de valeurs de teinte, de saturation et de valeur appartenant à l'arrière-plan).

A.6.1 Tests sur la base de données de Cornell

Pour comparer les différentes approches de segmentation d'images, nous avons utilisé la base de données Cornell, qui comprend 244 scènes RGB-D (une scène aléatoire pour chaque type d'objet) [75]. Pour évaluer la qualité de la segmentation obtenue, nous avons utilisé une métrique appelée intersection sur union (IoU) ou indice de Jaccard. Cette métrique mesure le chevauchement entre le masque de segmentation prédit (A) et le masque de vérité terrain (B), et va de 0 (aucun chevauchement) à 1 (chevauchement parfait) A.3. Les scènes ont été segmentées manuellement et les résultats ont été comparés à ceux obtenus avec les algorithmes de segmentation (voir Fig. A.14).

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{A.3}$$

Un score IoU plus élevé indique une meilleure performance, car cela signifie que les masques prédits sont plus similaires aux masques réalisés manuellement. Les résultats des différentes méthodes sont détaillés dans le tableau A.1 et la figure A.15.

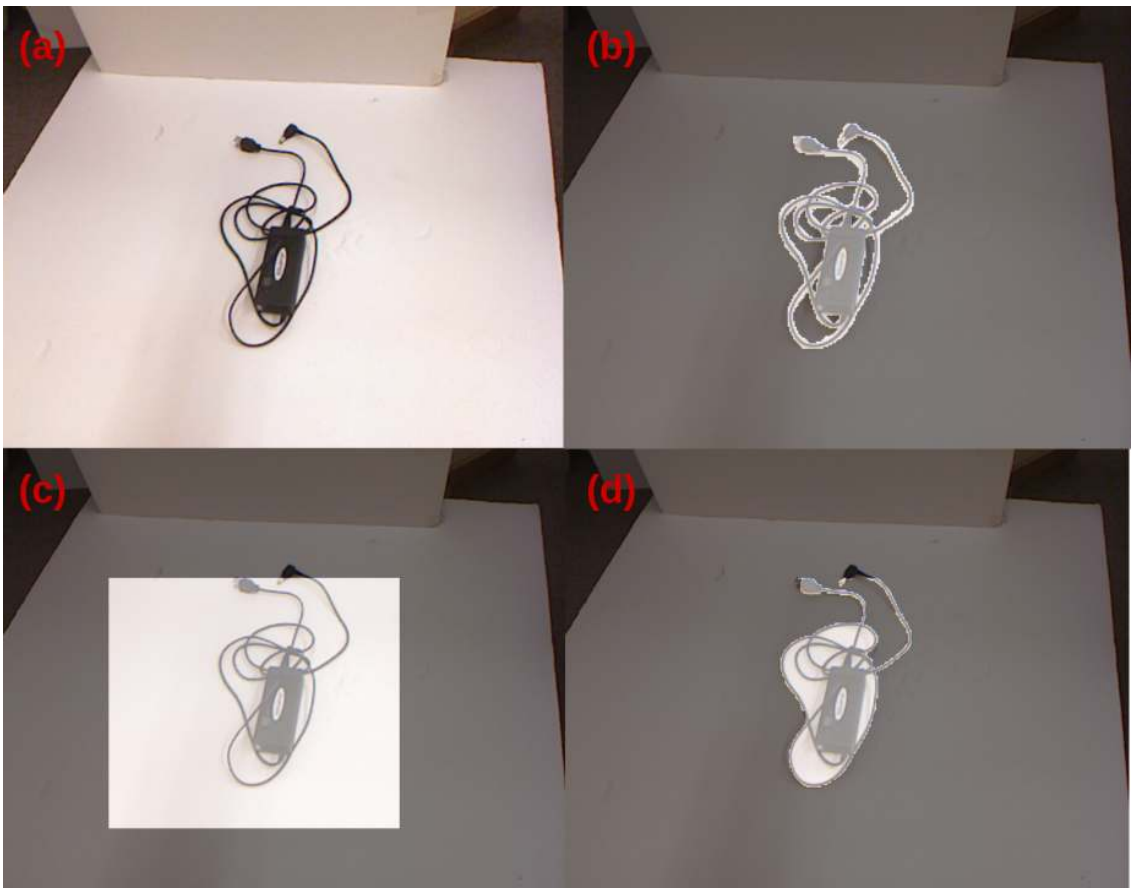


FIGURE A.14 – (a) l'image originale, (b) le masque extrait manuellement, (c) le masque croppant les bords et (d) le masque issu de la combinaison 'mixed' de segmentation.

Le tableau A.1 présente les résultats des différentes méthodes de segmentation appliquées aux images croppées (voir Fig. A.14 (c)) de la base de données Cornell [75].

Method	mean IoU	median IoU	mean time (s)
saliency	0.412	0.405	0.13
ransac+grabcut	0.600	0.723	2.72
closed_edges	0.614	0.653	0.08
mixed	0.627	0.650	0.11
background	0.689	0.739	0.09
hsv	0.696	0.779	0.07
saliency+grabcut	0.783	0.857	0.61
hsv+grabcut	0.797	0.865	0.43
background+grabcut	0.819	0.869	0.51
closed_edges+grabcut	0.826	0.874	0.47
mixed+grabcut	0.831	0.877	0.54

TABLE A.1 – Résultats des différentes méthodes de segmentation comparées aux segmentations manuelles de 244 scènes issues de la base de données Cornell [75]. Tous les algorithmes ont été appliqués sur des images recadrées (voir Fig. A.14 (c)). Les temps de la 3ème colonne correspondent aux temps de calcul moyen des segmentations réalisées avec un processeur Intel Core i7 8850h 2,60 GHz. Les meilleures performances en termes d'IoU (plus élevé indique une meilleure performance) sont obtenues en combinant les techniques de masquage « closed edges », « HSV » et « background », puis en passant le masque en entrée de l'algorithme GrabCut, obtenant un score moyen d'environ 0,83 pour un temps de calcul moyen par scène d'environ 0,53 secondes.

Les temps de calcul sont indiqués dans la troisième colonne et correspondent aux temps moyens de segmentation réalisés avec un CPU Intel Core i7 8850h 2.60GHz. Les résultats montrent que les méthodes de segmentation issues des données RGB, telles que 'background', 'HSV' et 'closed edges', obtiennent les meilleurs scores IoU lorsqu'elles sont combinées et suivies de l'algorithme GrabCut. En effet, cette combinaison permet d'obtenir un score moyen IoU de 0,83 pour un temps de calcul moyen par scène de 0,53 seconde.

La figure A.15 illustre la distribution des scores d'IoU obtenus par chaque méthode de segmentation. On peut y observer que les segmentations obtenant les meilleurs résultats proviennent des données RGB, plutôt que des méthodes utilisant les données de profondeur de la caméra (telles que ransac/depth).

En résumé, les résultats montrent que la combinaison des méthodes 'background', 'HSV' et 'closed edges' suivie de l'algorithme GrabCut est la plus performante pour la segmentation d'images de la base de données Cornell. Des approches utilisant l'apprentissage profond pourraient fournir de meilleurs résultats et l'une d'elle est brièvement abordée à la sous-section 3.4.6.

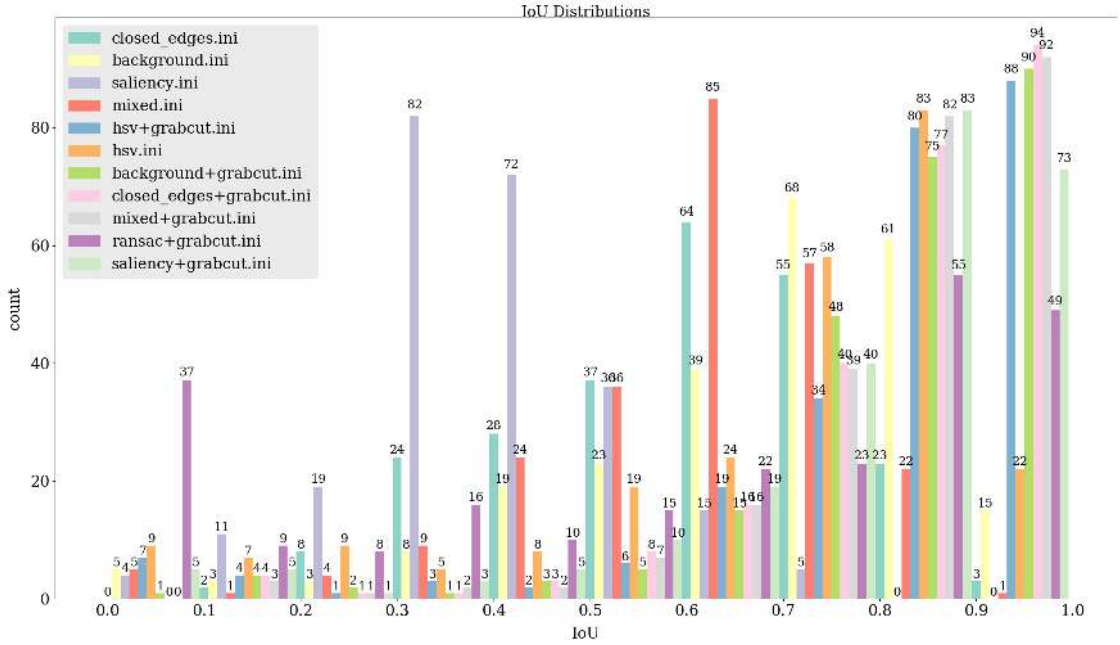


FIGURE A.15 – Résultats des différentes méthodes de segmentation comparés aux segmentations manuelles de 244 scènes issues de la base de données Cornell. Chaque histogramme représente la distribution des scores IoU par méthode, permettant de visualiser la performance des différents algorithmes de segmentation. Par exemple, pour la méthode HSV, on peut observer qu'il y a 9 segmentations ayant un score compris entre 0 et 0,1. Les résultats montrent que les méthodes de segmentation basées sur les données RGB, telles que « background », « HSV » et « closed edges », obtiennent les meilleurs scores IoU. En revanche, les méthodes utilisant les données de profondeur de la caméra, telles que RANSAC/profondeur, obtiennent des résultats moins performants.

Bibliographie

[1] Eman Ahmed, Alexandre Saint, Abd El Rahman Shabayek, Kseniya Cherenkova, Rig Das, Gleb Gusev, Djamila Aouada et Björn E. Ottersten. « Deep Learning Advances on Different 3D Data Representations : A Survey ». In : *CoRR* abs/1808.01462 (2018). arXiv : 1808.01462. URL : http://arxiv.org/abs/1808.01462.

[2] Mihai Andries, Yoann Fleytoux, J.-B. Mouret et Serena Ivaldi. « AGOD-Grasp : an Automatically Generated Object Dataset for benchmarking and training robotic grasping algorithms ». working paper or preprint. Juil. 2020. URL : https://hal.inria.fr/hal-03983079.

[3] Rika Antonova, Mia Kokic, Johannes A. Stork et Danica Kragic. « Global Search with Bernoulli Alternation Kernel for Task-oriented Grasping Informed by Simulation ». In : *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*. T. 87. Proceedings of Machine Learning Research. PMLR, 2018, p. 641-650. URL : http://proceedings.mlr.press/v87/antonova18a.html.

[4] Umar Asif, Jianbin Tang et Stefan Harrer. « GraspNet : An Efficient Convolutional Neural Network for Real-time Grasp Detection for Low-powered Devices ». In : *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. Sous la dir. de Jérôme Lang. ijcai.org, 2018, p. 4875-4882. DOI : 10.24963/ijcai.2018/677. URL : https://doi.org/10.24963/ijcai.2018/677.

[5] Yahav Avigal, Vishal Satish, Zachary Tam, Huang Huang, Harry Zhang, Michael Danielczuk, Jeffrey Ichnowski et Ken Goldberg. « AVPLUG : Approach Vector PLanning for Unicontact Grasping amid Clutter ». In : *17th IEEE International Conference on Automation Science and Engineering, CASE 2021, Lyon, France, August 23-27, 2021*. IEEE, 2021, p. 1140-1147. DOI : 10.1109/CASE49439.2021.9551652. URL : https://doi.org/10.1109/CASE49439.2021.9551652.

[6] Vijay Badrinarayanan, Alex Kendall et Roberto Cipolla. « SegNet : A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12 (2017), p. 2481-2495. DOI : 10.1109/TPAMI.2016.2644615. URL : https://doi.org/10.1109/TPAMI.2016.2644615.

[7] Ainur Begalinova. « Approaches for Intelligent Robot Grasping and Manipulation via Human Demonstration ». In : (2020). URL : https://pure.manchester.ac.uk/ws/portalfiles/portal/173343915/FULL_TEXT.PDF.

[8] Lars Berscheid, Christian Friedrich et Torsten Kröger. « Robot Learning of 6 DoF Grasping using Model-based Adaptive Primitives ». In : *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*. IEEE, 2021, p. 4474-4480. DOI : 10.1109/ICRA48506.2021.9560901. URL : https://doi.org/10.1109/ICRA48506.2021.9560901.

[9] Vishal Bhutani, Anima Majumder, Madhu Babu Vankadari, Samrat Dutta, Aaditya Asati et Swagat Kumar. « Attentive One-Shot Meta-Imitation Learning From Visual Demonstration ». In : *2022 International Conference on Robotics and Automation, ICRA 2022, Philadelphia, PA, USA, May 23-27, 2022*. IEEE, 2022, p. 8584-8590. DOI : 10.1109/ICRA46639.2022.9812281. URL : https://doi.org/10.1109/ICRA46639.2022.9812281.

[10] Asmita Singh Bisen et Himanshu Payal. « Collaborative robots for industrial tasks : A review ». In : *Materials Today : Proceedings* 52 (2022), p. 500-504.

[11] Jeannette Bohg, Antonio Morales, Tamim Asfour et Danica Kragic. « Data-Driven Grasp Synthesis - A Survey ». In : *IEEE Trans. Robotics* 30.2 (2014), p. 289-309. DOI : 10.1109/TRO.2013.2289018. URL : https://doi.org/10.1109/TRO.2013.2289018.

[12] Richard Bormann, Bruno Ferreira de Brito, Jochen Lindermayr, Marco Omainska et Mayank Patel. « Towards Automated Order Picking Robots for Warehouses and Retail ». In : *Computer Vision Systems, 12th International Conference, ICVS 2019, Thessaloniki, Greece, September 23-25, 2019, Proceedings*. Sous la dir. de Dimitrios Tzovaras, Dimitrios Giakoumis, Markus Vincze et Antonis A. Argyros. T. 11754. Lecture Notes in Computer Science. Springer, 2019, p. 185-198. DOI : 10.1007/978-3-030-34995-0_18. URL : https://doi.org/10.1007/978-3-030-34995-0%5C_18.

[13] Fabrizio Bottarel, Giulia Vezzani, Ugo Pattacini et Lorenzo Natale. « GRASPA 1.0 : GRASPA is a Robot Arm graSping Performance BenchmArk ». In : *IEEE Robotics and Automation Letters* 5.2 (2020), p. 836-843. DOI : 10.1109/LRA.2020.2965865. URL : https://doi.org/10.1109/LRA.2020.2965865.

[14] Abdeslam Boularias, James Andrew Bagnell et Anthony Stentz. « Learning to Manipulate Unknown Objects in Clutter by Reinforcement ». In : *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. Sous la dir. de Blai Bonet et Sven Koenig. AAAI Press, 2015, p. 1336-1342. URL : http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9360.

[15] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, Sergey Levine et Vincent Vanhoucke. « Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping ». In : *IEEE International Conference on Robotics and Automation (ICRA)*. 2018, p. 4243-4250.

[16] Samarth Brahmbhatt, Cusuh Ham, Charles C. Kemp et James Hays. « ContactDB : Analyzing and Predicting Grasp Contact via Thermal Imaging ». In : *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, p. 8709-8719. doi : 10.1109/CVPR.2019.00891. url : http://openaccess.thecvf.com/content_CVPR_2019/html/Brahmbhatt_ContactDB_Analyzing_and_Predicting_Grasp_Contact_via_Thermal_Imaging_CVPR_2019_paper.html.

[17] Romain Brégier, Frederic Devernay, Laetitia Leyrit et James L. Crowley. « Symmetry Aware Evaluation of 3D Object Detection and Pose Estimation in Scenes of Many Parts in Bulk ». In : *2017 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, p. 2209-2218. doi : 10.1109/ICCVW.2017.258. url : https://doi.org/10.1109/ICCVW.2017.258.

[18] Leo Breiman. « Random Forests ». In : *Machine learning* 45.1 (2001), p. 5-32. doi : 10.1023/A:1010933404324. url : https://doi.org/10.1023/A:1010933404324.

[19] Michel Breyer, Jen Jen Chung, Lionel Ott, Roland Siegwart et Juan I. Nieto. « Volumetric Grasping Network : Real-time 6 DOF Grasp Detection in Clutter ». In : *4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA*. Sous la dir. de Jens Kober, Fabio Ramos et Claire J. Tomlin. T. 155. Proceedings of Machine Learning Research. PMLR, 2020, p. 1602-1611. url : https://proceedings.mlr.press/v155/breyer21a.html.

[20] Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins et Alexander Lerchner. « Understanding disentangling in β-VAE ». In : *CoRR* abs/1804.03599 (2018). arXiv : 1804.03599. url : http://arxiv.org/abs/1804.03599.

[21] Junhao Cai, Jun Cen, Haokun Wang et Michael Yu Wang. « Real-Time Collision-Free Grasp Pose Detection With Geometry-Aware Refinement Using High-Resolution Volume ». In : *IEEE Robotics and Automation Letters* 7.2 (2022), p. 1888-1895. doi : 10.1109/LRA.2022.3142424.

[22] Roberto Calandra, Andrew Owens, Dinesh Jayaraman, Justin Lin, Wenzhen Yuan, Jitendra Malik, Edward H. Adelson et Sergey Levine. « More Than a Feeling : Learning to Grasp and Regrasp Using Vision and Touch ». In : *IEEE Robotics and Automation Letters* 3.4 (2018), p. 3300-3307. doi : 10.1109/LRA.2018.2852779. url : https://doi.org/10.1109/LRA.2018.2852779.

[23] Roberto Calandra, Andrew Owens, Manu Upadhyaya, Wenzhen Yuan, Justin Lin, Edward H. Adelson et Sergey Levine. « The Feeling of Success : Does Touch Sensing Help Predict Grasp Outcomes ? » In : *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*. T. 78. Proceedings of Machine Learning Research. PMLR, 2017, p. 314-323. url : http://proceedings.mlr.press/v78/calandra17a.html.

[24] Shehan Caldera, Alexander Rassau et Douglas Chai. « Review of Deep Learning Methods in Robotic Grasp Detection ». In : *Multimodal Technologies and Interaction* 2.3 (2018), p. 57. doi : 10.3390/mti2030057. url : https://doi.org/10.3390/mti2030057.

[25] Sylvain Calinon. « A tutorial on task-parameterized movement learning and retrieval ». In : *Intelligent Service Robotics* 9.1 (2016), p. 1-29. doi : 10.1007/s11370-015-0187-9. url : https://doi.org/10.1007/s11370-015-0187-9.

[26] Sylvain Calinon, Florent D'halluin, Eric L. Sauser, Darwin G. Caldwell et Aude Billard. « Learning and Reproduction of Gestures by Imitation ». In : *IEEE Robotics Automation Magazine* 17.2 (2010), p. 44-54. doi : 10.1109/MRA.2010.936947. url : https://doi.org/10.1109/MRA.2010.936947.

[27] Berk Çalli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha S. Srinivasa, Pieter Abbeel et Aaron M. Dollar. « Yale-CMU-Berkeley dataset for robotic manipulation research ». In : *International Journal of Robotics Research* 36.3 (2017), p. 261-268. doi : 10.1177/0278364917700714. url : https://doi.org/10.1177/0278364917700714.

[28] Berk Çalli, Aaron Walsman, Arjun Singh, Siddhartha S. Srinivasa, Pieter Abbeel et Aaron M. Dollar. « Benchmarking in Manipulation Research : The YCB Object and Model Set and Benchmarking Protocols ». In : *CoRR* abs/1502.03143 (2015). arXiv : 1502.03143. url : http://arxiv.org/abs/1502.03143.

[29] Berk Çalli, Aaron Walsman, Arjun Singh, Siddhartha S. Srinivasa, Pieter Abbeel et Aaron M. Dollar. « Benchmarking in Manipulation Research : Using the Yale-CMU-Berkeley Object and Model Set ». In : *IEEE Robotics Automation Magazine* 22.3 (2015), p. 36-52. doi : 10.1109/MRA.2015.2448951. url : https://doi.org/10.1109/MRA.2015.2448951.

[30] John Canny. « A computational approach to edge detection ». In : *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), p. 679-698.

[31] Hanwen Cao, Haoshu Fang, Wenhai Liu et Cewu Lu. « SuctionNet-1Billion : A Large-Scale Benchmark for Suction Grasping ». In : *IEEE Robotics and Automation Letters* 6.4 (2021), p. 8718-8725. doi : 10.1109/LRA.2021.3115406. url : https://doi.org/10.1109/LRA.2021.3115406.

[32] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi et Fisher Yu. « ShapeNet : An Information-Rich 3D Model Repository ». In : *CoRR* abs/1512.03012 (2015). arXiv : 1512.03012. url : http://arxiv.org/abs/1512.03012.

[33] Yevgen Chebotar, Karol Hausman, Oliver Kroemer, Gaurav S. Sukhatme et Stefan Schaal. « Regrasping Using Tactile Perception and Supervised Policy Learning ». In : *2017 AAAI Spring Symposia, Stanford University, Palo Alto, California, USA, March 27-29, 2017*. AAAI Press, 2017. url : http://aaai.org/ocs/index.php/SSS/SSS17/paper/view/15236.

[34] Liang-Chieh Chen, George Papandreou, Florian Schroff et Hartwig Adam. « Rethinking Atrous Convolution for Semantic Image Segmentation ». In : *CoRR* abs/1706.05587 (2017). arXiv : 1706.05587. URL : http://arxiv.org/abs/1706.05587.

[35] Ruijie Chen, Ning Gao, Ngo Anh Vien, Hanna Ziesche et Gerhard Neumann. « Meta-Learning Regrasping Strategies for Physical-Agnostic Objects ». In : *CoRR* abs/2205.11110 (2022). DOI : 10.48550/arXiv.2205.11110. arXiv : 2205.11110. URL : https://doi.org/10.48550/arXiv.2205.11110.

[36] Sachin Chitta, Ioan Sucan et Steve Cousins. « MoveIt ! » In : *IEEE Robotics & Automation Magazine* 19.1 (2012), p. 18-19.

[37] Young Sang Choi, Travis Deyle, Tiffany Chen, Jonathan D Glass et Charles C Kemp. « A list of household objects for robotic retrieval prioritized by people with ALS ». In : *2009 IEEE International Conference on Rehabilitation Robotics*. IEEE. 2009, p. 510-517.

[38] Yunho Choi, Hogun Kee, Kyungjae Lee, Jaegoo Choy, Junhong Min, Sohee Lee et Songhwai Oh. « Hierarchical 6-DoF Grasping with Approaching Direction Selection ». In : *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*. IEEE, 2020, p. 1553-1559. DOI : 10.1109/ICRA40945.2020.9196678. URL : https://doi.org/10.1109/ICRA40945.2020.9196678.

[39] Fu-Jen Chu, Ruinian Xu et Patricio A. Vela. « Deep Grasp : Detection and Localization of Grasps with Deep Neural Networks ». In : *CoRR* abs/1802.00520 (2018). arXiv : 1802.00520. URL : http://arxiv.org/abs/1802.00520.

[40] Fu-Jen Chu, Ruinian Xu et Patricio A. Vela. « Real-World Multiobject, Multigrasp Detection ». In : *IEEE Robotics and Automation Letters* 3.4 (2018), p. 3355-3362. DOI : 10.1109/LRA.2018.2852777. URL : https://doi.org/10.1109/LRA.2018.2852777.

[41] Nikolaus Correll, Kostas E. Bekris, Dmitry Berenson, Oliver Brock, Albert J. Causo, Kris Hauser, Kei Okada, Alberto Rodriguez, Joseph M. Romano et Peter R. Wurman. « Analysis and Observations From the First Amazon Picking Challenge ». In : *IEEE Transactions on Automation Science and Engineering* 15.1 (2018), p. 172-188. DOI : 10.1109/TASE.2016.2600527. URL : https://doi.org/10.1109/TASE.2016.2600527.

[42] Corinna Cortes et Vladimir Vapnik. « Support-vector networks ». In : *Machine learning* 20.3 (1995), p. 273-297.

[43] Mark R. Cutkosky. « On grasp choice, grasp models, and the design of hands for manufacturing tasks ». In : *IEEE Transactions on robotics and automation* 5.3 (1989), p. 269-279. DOI : 10.1109/70.34763. URL : https://doi.org/10.1109/70.34763.

[44] Hal Dardick. « Mechanical Advantage - Two Northwestern University engineers are developing cobots – machines that, unlike robots, cooperate with workers without displacing them ». In : *Chicago Tribune* (oct. 1996).

[45] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine et Chelsea Finn. « RoboNet : Large-Scale Multi-Robot Learning ». In : *3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings*. Sous la dir. de Leslie Pack Kaelbling, Danica Kragic et Komei Sugiura. T. 100. Proceedings of Machine Learning Research. PMLR, 2019, p. 885-897. URL : http://proceedings.mlr.press/v100/dasari20a.html.

[46] Google Brain Robotics Data. *Procedurally generated random objects*. URL : https://sites.google.com/site/brainrobotdata/home/models.

[47] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li et Li Fei-Fei. « Imagenet : A large-scale hierarchical image database ». In : *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, p. 248-255.

[48] Emily Denton et Rob Fergus. « Stochastic Video Generation with a Learned Prior ». In : *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Sous la dir. de Jennifer G. Dy et Andreas Krause. T. 80. Proceedings of Machine Learning Research. PMLR, 2018, p. 1182-1191. URL : http://proceedings.mlr.press/v80/denton18a.html.

[49] Amaury Depierre, Emmanuel Dellandréa et Liming Chen. « Jacquard : A Large Scale Dataset for Robotic Grasp Detection ». In : *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018*. IEEE, 2018, p. 3511-3516. DOI : 10.1109/IROS.2018.8593950. URL : https://doi.org/10.1109/IROS.2018.8593950.

[50] Mehmet Remzi Dogar et Siddhartha S. Srinivasa. « A Planning Framework for Non-Prehensile Manipulation under Clutter and Uncertainty ». In : *Auton. Robots* 33.3 (2012), p. 217-236. DOI : 10.1007/s10514-012-9306-z. URL : https://doi.org/10.1007/s10514-012-9306-z.

[51] Harris Drucker. « Improving Regressors using Boosting Techniques ». In : *Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, Tennessee, USA, July 8-12, 1997*. Sous la dir. de Douglas H. Fisher. Morgan Kaufmann, 1997, p. 107-115.

[52] Royson Dsouza. « The Art of Tactile Sensing : A State of Art Survey ». In : *International Journal of Sciences : Basic and Applied Research (IJSBAR)* 26 (mai 2016), p. 252-266.

[53] Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex X. Lee et Sergey Levine. « Visual Foresight : Model-Based Deep Reinforcement Learning for Vision-Based Robotic Control ». In : *CoRR* abs/1812.00568 (2018). arXiv : 1812.00568. URL : http://arxiv.org/abs/1812.00568.

[54] Clemens Eppner, Arsalan Mousavian et Dieter Fox. « ACRONYM : A Large-Scale Grasp Dataset Based on Simulation ». In : *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*. IEEE, 2021, p. 6222-6227. DOI : 10.1109/ICRA48506.2021.9560844. URL : https://doi.org/10.1109/ICRA48506.2021.9560844.

[55] Carlo Ferrari et John F. Canny. « Planning optimal grasps ». In : *IEEE International Conference on Robotics and Automation (ICRA)*. 1992, p. 2290-2295.

[56] Yoann Fleytoux, Anji Ma, Serena Ivaldi et Jean-Baptiste Mouret. « Data-efficient learning of object-centric grasp preferences ». In : *2022 International Conference on Robotics and Automation, ICRA 2022, Philadelphia, PA, USA, May 23-27, 2022*. IEEE, 2022, p. 6337-6343. doi : 10.1109/ICRA46639.2022.9811760. url : https://doi.org/10.1109/ICRA46639.2022.9811760.

[57] James J Gibson. « The theory of affordances ». In : *Hilldale, USA* 1.2 (1977), p. 67-82.

[58] Nikolaos Gkanatsios, Georgia Chalvatzaki, Petros Maragos et Jan Peters. « Orientation Attentive Robot Grasp Synthesis ». In : *CoRR* abs/2006.05123 (2020). arXiv : 2006.05123. url : https://arxiv.org/abs/2006.05123.

[59] Raymond C Goertz. « Remote-control manipulator ». Brev. US2632574A. 1949. url : https://patents.google.com/patent/US2632574.

[60] Marcus Gualtieri et Robert Platt Jr. « Learning 6-DoF Grasping and Pick-Place Using Attention Focus ». In : *2nd Annual Conference on Robot Learning (CoRL)*. T. 87. PMLR, 2018, p. 477-486.

[61] Saurabh Gupta, Ross B. Girshick, Pablo Andrés Arbeláez et Jitendra Malik. « Learning Rich Features from RGB-D Images for Object Detection and Segmentation ». In : *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII*. Sous la dir. de David J. Fleet, Tomás Pajdla, Bernt Schiele et Tinne Tuytelaars. T. 8695. Lecture Notes in Computer Science. Springer, 2014, p. 345-360. doi : 10.1007/978-3-319-10584-0_23. url : https://doi.org/10.1007/978-3-319-10584-0%5C_23.

[62] Georg Halmetschlager-Funek, Markus Suchi, Martin Kampel et Markus Vincze. « An Empirical Evaluation of Ten Depth Cameras : Bias, Precision, Lateral Noise, Different Lighting Conditions and Materials, and Multiple Sensor Setups in Indoor Environments ». In : *IEEE Robotics Automation Magazine* 26.1 (2019), p. 67-77. doi : 10.1109/MRA.2018.2852795. url : https://doi.org/10.1109/MRA.2018.2852795.

[63] Kaiming He, Xiangyu Zhang, Shaoqing Ren et Jian Sun. « Deep Residual Learning for Image Recognition ». In : *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, p. 770-778. doi : 10.1109/CVPR.2016.90. url : https://doi.org/10.1109/CVPR.2016.90.

[64] François Hélénon, Laurent Bimont, Eric Nyiri, Stéphane Thiery et Olivier Gibaru. « Learning prohibited and authorised grasping locations from a few demonstrations ». In : *29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. 2020, p. 1094-1100.

[65] James Hensman, Nicoló Fusi et Neil D. Lawrence. « Gaussian Processes for Big Data ». In : *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI 2013, Bellevue, WA, USA, August 11-15, 2013*. Sous la dir. d'Ann E. Nicholson et Padhraic Smyth. AUAI Press, 2013. URL : https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1%5C&smnu=2%5C&article%5C_id=2389%5C&proceeding%5C_id=29.

[66] James Hensman, Alexander G. de G. Matthews et Zoubin Ghahramani. « Scalable Variational Gaussian Process Classification ». In : *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS*. T. 38. 2015.

[67] Irina Higgins, Loic Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed et Alexander Lerchner. « beta-VAE : Learning Basic Visual Concepts with a Constrained Variational Framework ». In : *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL : https://openreview.net/forum?id=Sy2fzU9gl.

[68] Geoffrey E Hinton. « Connectionist learning procedures ». In : *Machine learning*. Elsevier, 1990, p. 555-610.

[69] Tomas Hodan, Pavel Haluza, Stepán Obdrzálek, Jiri Matas, Manolis I. A. Lourakis et Xenophon Zabulis. « T-LESS : An RGB-D Dataset for 6D Pose Estimation of Texture-Less Objects ». In : *2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017, Santa Rosa, CA, USA, March 24-31, 2017*. IEEE Computer Society, 2017, p. 880-888. DOI : 10.1109/WACV.2017.103. URL : https://doi.org/10.1109/WACV.2017.103.

[70] Xiaodi Hou et Liqing Zhang. « Saliency Detection : A Spectral Residual Approach ». In : *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. 2007.

[71] Yongqiang Huang, Matteo Bianchi, Minas V. Liarokapis et Yu Sun. « Recent Data Sets on Object Manipulation : A Survey ». In : *Big Data* 4.4 (2016), p. 197-216. DOI : 10.1089/big.2016.0042. URL : https://doi.org/10.1089/big.2016.0042.

[72] Jeffrey Ichnowski, Yahav Avigal, Justin Kerr et Ken Goldberg. « DexNeRF : Using a Neural Radiance Field to Grasp Transparent Objects ». In : *Conference on Robot Learning, 8-11 November 2021, London, UK*. Sous la dir. d'Aleksandra Faust, David Hsu et Gerhard Neumann. T. 164. Proceedings of Machine Learning Research. PMLR, 2021, p. 526-536. URL : https://proceedings.mlr.press/v164/ichnowski22a.html.

[73] Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell et Konstantinos Bousmalis. « Sim-To-Real via Sim-To-Sim : Data-Efficient Robotic Grasping via Randomized-To-Canonical Adaptation Networks ». In : *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, p. 12627-12637. DOI : 10.1109/CVPR.

2019.01291. URL : http://openaccess.thecvf.com/content_CVPR_2019/html/James_Sim-To-Real_via_Sim-To-Sim_Data-Efficient_Robotic_Grasping_via_Randomized-To-Canonical_Adaptation_Networks_CVPR_2019_paper.html.

[74] Ping JIANG, Junji OAKI, Yoshiyuki ISHIHARA, Jun'ichiro OOGA, Haifeng HAN, Atsushi SUGAHARA, Seiji TOKURA, Haruna ETO, Kazuma KOMODA et Akihito OGAWA. « Learning Suction Graspability Considering Grasp Quality and Robot Reachability for Bin-Picking ». In : *Frontiers Neurorobotics* 16 (2022), p. 806898. DOI : 10.3389/fnbot.2022.806898. URL : https://doi.org/10.3389/fnbot.2022.806898.

[75] Yun JIANG, Stephen MOSESON et Ashutosh SAXENA. « Efficient grasping from RGBD images : Learning using a new rectangle representation ». In : *IEEE International Conference on Robotics and Automation (ICRA)*. 2011.

[76] Zhenyu JIANG, Yifeng ZHU, Maxwell SVETLIK, Kuan FANG et Yuke ZHU. « Synergies Between Affordance and Geometry : 6-DoF Grasp Detection via Implicit Representations ». In : *Robotics : Science and Systems XVII, Virtual Event, July 12-16, 2021*. Sous la dir. de Dylan A. SHELL, Marc TOUSSAINT et M. Ani HSIEH. 2021. DOI : 10.15607/RSS.2021.XVII.024. URL : https://doi.org/10.15607/RSS.2021.XVII.024.

[77] Edward JOHNS, Stefan LEUTENEGGER et Andrew J. DAVISON. « Deep learning a grasp function for grasping under gripper pose uncertainty ». In : *IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. IEEE, 2016, p. 4461-4468. DOI : 10.1109/IROS.2016.7759657. URL : https://doi.org/10.1109/IROS.2016.7759657.

[78] Pakorn KAEWTRAKULPONG et Richard BOWDEN. « An improved adaptive background mixture model for real-time tracking with shadow detection ». In : *Video-based surveillance systems : Computer vision and distributed processing* (2002), p. 135-144.

[79] Dmitry KALASHNIKOV, Alex IRPAN, Peter PASTOR, Julian IBARZ, Alexander HERZOG, Eric JANG, Deirdre QUILLEN, Ethan HOLLY, Mrinal KALAKRISHNAN, Vincent VANHOUCKE et Sergey LEVINE. « QT-Opt : Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation ». In : *CoRR* abs/1806.10293 (2018). arXiv : 1806.10293. URL : http://arxiv.org/abs/1806.10293.

[80] Hamidreza KASAEI et Mohammadreza KASAEI. « MVGrasp : Real-time multi-view 3D object grasping in highly cluttered environments ». In : *Robotics and Autonomous Systems* 160 (2023), p. 104313. DOI : 10.1016/j.robot.2022.104313. URL : https://doi.org/10.1016/j.robot.2022.104313.

[81] Alexander KASPER, Zhixing XUE et Rüdiger DILLMANN. « The KIT object models database : An object model database for object recognition, localization and manipulation in service robotics ». In : *International Journal of Robotics Research* 31.8 (2012), p. 927-934. DOI : 10.1177/0278364912445831. URL : https://doi.org/10.1177/0278364912445831.

[82] Diederik P. Kingma et Max Welling. « Auto-Encoding Variational Bayes ». In : *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Sous la dir. d'Yoshua Bengio et Yann LeCun. 2014. URL : http://arxiv.org/abs/1312.6114.

[83] Durk P Kingma, Tim Salimans et Max Welling. « Variational Dropout and the Local Reparameterization Trick ». In : *Advances in Neural Information Processing Systems*. Sous la dir. de C. Cortes, N. Lawrence, D. Lee, M. Sugiyama et R. Garnett. T. 28. Curran Associates, Inc., 2015. URL : https://proceedings.neurips.cc/paper_files/paper/2015/file/bc7316929fe1545bf0b98d114ee3ecb8-Paper.pdf.

[84] Thomas N. Kipf et Max Welling. « Semi-Supervised Classification with Graph Convolutional Networks ». In : *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL : https://openreview.net/forum?id=SJU4ayYgl.

[85] Hedvig Kjellström, Javier Romero et Danica Kragic. « Visual object-action recognition : Inferring object affordances from human demonstration ». In : *Computer Vision and Image Understanding* 115.1 (2011), p. 81-90. DOI : 10.1016/j.cviu.2010.08.002. URL : https://doi.org/10.1016/j.cviu.2010.08.002.

[86] Kilian Kleeberger, Richard Bormann, Werner Kraus et Marco F Huber. « A survey on learning-based robotic grasping ». In : *Current Robotics Reports* 1.4 (2020), p. 239-249.

[87] Iasonas Kokkinos, Michael Bronstein et al. « Dense scale invariant descriptors for images and surfaces ». Thèse de doct. INRIA, 2012.

[88] Alex Krizhevsky, Ilya Sutskever et Geoffrey E. Hinton. « ImageNet Classification with Deep Convolutional Neural Networks ». In : *Advances in Neural Information Processing Systems 25 : 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*. Sous la dir. de Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou et Kilian Q. Weinberger. 2012, p. 1106-1114. URL : https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html.

[89] Sulabh Kumra, Shirin Joshi et Ferat Sahin. « Antipodal Robotic Grasping using Generative Residual Convolutional Neural Network ». In : *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021*. IEEE, 2020, p. 9626-9633. DOI : 10.1109/IROS45743.2020.9340777. URL : https://doi.org/10.1109/IROS45743.2020.9340777.

[90] Sulabh Kumra et Christopher Kanan. « Robotic grasp detection using deep convolutional neural networks ». In : *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*. IEEE, 2017, p. 769-776. DOI : 10.1109/IROS.2017.8202237. URL : https://doi.org/10.1109/IROS.2017.8202237.

[91] Kevin Lai, Liefeng Bo, Xiaofeng Ren et Dieter Fox. « A large-scale hierarchical multi-view RGB-D object dataset ». In : *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*. IEEE, 2011, p. 1817-1824. DOI : 10.1109/ICRA.2011.5980382. URL : https://doi.org/10.1109/ICRA.2011.5980382.

[92] Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian H. Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, Dinesh Jayaraman et Roberto Calandra. « DIGIT : A Novel Design for a Low-Cost Compact High-Resolution Tactile Sensor With Application to In-Hand Manipulation ». In : *IEEE Robotics and Automation Letters* 5.3 (2020), p. 3838-3845. DOI : 10.1109/LRA.2020.2977257. URL : https://doi.org/10.1109/LRA.2020.2977257.

[93] Mike Lambeta, Huazhe Xu, Jingwei Xu, Po-Wei Chou, Shaoxiong Wang, Trevor Darrell et Roberto Calandra. « PyTouch : A Machine Learning Library for Touch Processing ». In : *IEEE International Conference on Robotics and Automation (ICRA)* (2021). URL : https://arxiv.org/abs/2105.12791.

[94] Ta-Chih Lee, Rangasami L. Kashyap et Chong-Nam Chu. « Building Skeleton Models via 3-D Medial Surface/Axis Thinning Algorithms ». In : *Graphical Models and Image Processing* 56.6 (1994), p. 462-478.

[95] Sylvain Lefebvre, Samuel Hornus, Fabrice Neyret et al. « Octree textures on the GPU ». In : *GPU gems* 2 (2005), p. 595-613.

[96] Ian Lenz, Honglak Lee et Ashutosh Saxena. « Deep Learning for Detecting Robotic Grasps ». In : *Robotics : Science and Systems IX*. T. 34. 4-5. 2013, p. 705-724. DOI : 10.1177/0278364914549607. URL : https://doi.org/10.1177/0278364914549607.

[97] Ian Lenz, Honglak Lee et Ashutosh Saxena. « Deep learning for detecting robotic grasps ». In : *International Journal of Robotics Research* 34.4-5 (2015), p. 705-724. DOI : 10.1177/0278364914549607. URL : https://doi.org/10.1177/0278364914549607.

[98] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz et Deirdre Quillen. « Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection ». In : *International Journal of Robotics Research* 37.4-5 (2018), p. 421-436.

[99] Hongzhuo Liang, Xiaojian Ma, Shuang Li, Michael Görner, Song Tang, Bin Fang, Fuchun Sun et Jianwei Zhang. « PointNetGPD : Detecting Grasp Configurations from Point Sets ». In : *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*. IEEE, 2019, p. 3629-3635. DOI : 10.1109/ICRA.2019.8794435. URL : https://doi.org/10.1109/ICRA.2019.8794435.

[100] Qingquan Lin et Dan Chen. « Target recognition and optimal grasping based on deep learning ». In : *2018 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*. IEEE. 2018, p. 1-6.

[101] Weiyu Liu, Angel Andres Daruna et Sonia Chernova. « CAGE : Context-Aware Grasping Engine ». In : *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*. IEEE, 2020, p. 2550-2556. doi : 10.1109/ICRA40945.2020.9197289. url : https://doi.org/10.1109/ICRA40945.2020.9197289.

[102] Jonathan Long, Evan Shelhamer et Trevor Darrell. « Fully convolutional networks for semantic segmentation ». In : *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. IEEE Computer Society, 2015, p. 3431-3440. doi : 10.1109/CVPR.2015.7298965. url : https://doi.org/10.1109/CVPR.2015.7298965.

[103] Xibai Lou, Yang Yang et Changhyun Choi. « Collision-Aware Target-Driven Object Grasping in Constrained Environments ». In : *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*. IEEE, 2021, p. 6364-6370. doi : 10.1109/ICRA48506.2021.9561473. url : https://doi.org/10.1109/ICRA48506.2021.9561473.

[104] Ulrike von Luxburg. « A tutorial on spectral clustering ». In : *Statistics and Computing* 17.4 (2007), p. 395-416. doi : 10.1007/s11222-007-9033-z. url : https://doi.org/10.1007/s11222-007-9033-z.

[105] Anji Ma, Yoann Fleytoux, Jean-Baptiste Mouret et Serena Ivaldi. « VP-GO : A 'Light' Action-Conditioned Visual Prediction Model for Grasping Objects ». In : *International Conference on Advanced Robotics and Mechatronics , ICARM 2022, Guilin, China, July 9-11, 2022*. IEEE, 2022, p. 37-44. doi : 10.1109/ICARM54641.2022.9959321. url : https://doi.org/10.1109/ICARM54641.2022.9959321.

[106] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea et Ken Goldberg. « Dex-Net 2.0 : Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics ». In : *Robotics : Science and Systems XIII*. 2017. doi : 10.15607/RSS.2017.XIII.058.

[107] Jeffrey Mahler, Matthew Matl, Xinyu Liu, Albert Li, David V. Gealy et Ken Goldberg. « Dex-Net 3.0 : Computing Robust Vacuum Suction Grasp Targets in Point Clouds Using a New Analytic Model and Deep Learning ». In : *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*. IEEE, 2018, p. 1-8. doi : 10.1109/ICRA.2018.8460887. url : https://doi.org/10.1109/ICRA.2018.8460887.

[108] Jeffrey Mahler, Florian T. Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kröger, James J. Kuffner et Ken Goldberg. « Dex-Net 1.0 : A cloud-based network of 3D objects for robust grasp planning using a Multi-Armed Bandit model with correlated rewards ». In : *IEEE International Conference on Robotics and Automation, (ICRA)*. 2016.

[109] J. Makhoul. « Linear prediction : A tutorial review ». In : *Proceedings of the IEEE* 63.4 (1975), p. 561-580. doi : 10.1109/PROC.1975.9792.

[110] Xanthippi Markenscoff et Christos H. Papadimitriou. « Optimum Grip of a Polygon ». In : *International Journal of Robotics Research* 8.2 (1989), p. 17-29. doi : 10.1177/027836498900800202. url : https://doi.org/10.1177/027836498900800202.

[111] Haggai Maron, Meirav Galun, Noam Aigerman, Miri Trope, Nadav Dym, Ersin Yumer, Vladimir G. Kim et Yaron Lipman. « Convolutional neural networks on surfaces via seamless toric covers ». In : *ACM Transactions on Graphics* 36.4 (2017), 71 :1-71 :10. doi : 10.1145/3072959.3073616. url : https://doi.org/10.1145/3072959.3073616.

[112] Mauricio Matamoros, Caleb Rascon, Sven Wachsmuth, Alexander William Moriarty, Johannes Kummert, Justin Hart, Sammy Pfeiffer, Matthijs van der Brugh et Maxime St-Pierre. *RoboCup@Home 2019 : Rules and Regulations (draft)*. http://www.robocupathome.org/rules/2019_rulebook.pdf. 2019.

[113] Matthew Matl, Vishal Satish, Michael Danielczuk, Bill DeRose, Stephen McKinley et Ken Goldberg. « Learning ambidextrous robot grasping policies ». In : *Sci. Robotics* 4.26 (2019). doi : 10.1126/scirobotics.aau4984. url : https://doi.org/10.1126/scirobotics.aau4984.

[114] Daniel Maturana et Sebastian A. Scherer. « VoxNet : A 3D Convolutional Neural Network for real-time object recognition ». In : *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015, Hamburg, Germany, September 28 - October 2, 2015*. IEEE, 2015, p. 922-928. doi : 10.1109/IROS.2015.7353481. url : https://doi.org/10.1109/IROS.2015.7353481.

[115] Robin McKie. « Sellafield : the most hazardous place in Europe ». In : *The Guardian* 19 (2009). Science Editor. url : https://www.theguardian.com/environment/2009/apr/19/sellafield-nuclear-plant-cumbria-hazards.

[116] Pieter Van Molle, Tim Verbelen, Elias De Coninck, Cedric De Boom, Pieter Simoens et Bart Dhoedt. « Learning to Grasp from a Single Demonstration ». In : *CoRR* abs/1806.03486 (2018). arXiv : 1806.03486. url : http://arxiv.org/abs/1806.03486.

[117] LIVRE Gareth J. Monkman, Stefan Hesse, Ralf Steinmann et Henrik Schunk. *Robot Grippers*. Wiley-VCH, 2007. url : https://www.wiley.com/en-us/Robot+Grippers-p-9783527609895.

[118] Douglas Morrison, Peter Corke et Jürgen Leitner. « EGAD ! An Evolved Grasping Analysis Dataset for Diversity and Reproducibility in Robotic Manipulation ». In : *IEEE Robotics and Automation Letters* 5.3 (2020), p. 4368-4375. doi : 10.1109/LRA.2020.2992195. url : https://doi.org/10.1109/LRA.2020.2992195.

[119] Douglas Morrison, Peter Corke et Jürgen Leitner. « Learning robust, real-time, reactive robotic grasping ». In : *International Journal of Robotics Research* 39.2-3 (2020). doi : 10.1177/0278364919859066. url : https://doi.org/10.1177/0278364919859066.

[120] Douglas Morrison, Juxi Leitner et Peter Corke. « Closing the Loop for Robotic Grasping : A Real-time, Generative Grasp Synthesis Approach ». In : *Robotics : Science and Systems XIV*. 2018.

[121] Jean-Baptiste Mouret et Jeff Clune. « Illuminating search spaces by mapping elites ». In : *CoRR* abs/1504.04909 (2015). arXiv : 1504.04909. URL : http://arxiv.org/abs/1504.04909.

[122] Arsalan Mousavian, Clemens Eppner et Dieter Fox. « 6-DOF GraspNet : Variational Grasp Generation for Object Manipulation ». In : *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2019, p. 2901-2910. DOI : 10.1109/ICCV.2019.00299. URL : https://doi.org/10.1109/ICCV.2019.00299.

[123] Mario Rios Muñoz, Lambert Schomaker et S Hamidreza Kasaei. « Extending GG-CNN through Automated Model Space Exploration using Knowledge Transfer ». In : ().

[124] Adithyavairavan Murali, Weiyu Liu, Kenneth Marino, Sonia Chernova et Abhinav Gupta. « Same Object, Different Grasps : Data and Semantic Knowledge for Task-Oriented Grasping ». In : *4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA*. Sous la dir. de Jens Kober, Fabio Ramos et Claire J. Tomlin. T. 155. Proceedings of Machine Learning Research. PMLR, 2020, p. 1540-1557. URL : https://proceedings.mlr.press/v155/murali21a.html.

[125] Adithyavairavan Murali, Arsalan Mousavian, Clemens Eppner, Chris Paxton et Dieter Fox. « 6-DOF Grasping for Target-driven Object Manipulation in Clutter ». In : *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*. IEEE, 2020, p. 6232-6238. DOI : 10.1109/ICRA40945.2020.9197318. URL : https://doi.org/10.1109/ICRA40945.2020.9197318.

[126] Austin Myers, Angjoo Kanazawa, Cornelia Fermuller et Yiannis Aloimonos. « Affordance of object parts from geometric features ». In : *Workshop on Vision meets Cognition, CVPR*. T. 9. 2014.

[127] Austin Myers, Ching L Teo, Cornelia Fermüller et Yiannis Aloimonos. « Affordance detection of tool parts from geometric features ». In : *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, p. 1374-1381.

[128] Ashvin Nair, Dian Chen, Pulkit Agrawal, Phillip Isola, Pieter Abbeel, Jitendra Malik et Sergey Levine. « Combining self-supervised learning and imitation for vision-based rope manipulation ». In : *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*. IEEE, 2017, p. 2146-2153. DOI : 10.1109/ICRA.2017.7989247. URL : https://doi.org/10.1109/ICRA.2017.7989247.

[129] Van-Duc Nguyen. « Constructing Force-Closure Grasps ». In : *International Journal of Robotics Research* 7.3 (1988), p. 3-16. DOI : 10.1177/027836498800700301. URL : https://doi.org/10.1177/027836498800700301.

[130] Van-Duc NGUYEN. « Constructing force-closure grasps ». In : *Proceedings of the 1986 IEEE International Conference on Robotics and Automation, San Francisco, California, USA, April 7-10, 1986*. IEEE, 1986, p. 1368-1373. DOI : 10.1109/ROBOT.1986.1087483. URL : https://doi.org/10.1109/ROBOT.1986.1087483.

[131] LIVRE OECD. *OECD Science, Technology and Innovation Outlook 2016*. 2016, p. 192. DOI : https://doi.org/https://doi.org/10.1787/sti_in_outlook-2016-en. URL : https://www.oecd-ilibrary.org/content/publication/sti_in_outlook-2016-en.

[132] Damir OMRCEN, Christian BÖGE, Tamim ASFOUR, Ales UDE et Rüdiger DILLMANN. « Autonomous acquisition of pushing actions to support object grasping with a humanoid robot ». In : *9th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2009, Paris, France, December 7-10, 2009*. IEEE, 2009, p. 277-283. DOI : 10.1109/ICHR.2009.5379566. URL : https://doi.org/10.1109/ICHR.2009.5379566.

[133] Andrej ORSULA, Simon BØGH, Miguel A. OLIVARES-MÉNDEZ et Carol MARTINEZ. « Learning to Grasp on the Moon from 3D Octree Observations with Deep Reinforcement Learning ». In : *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2022, Kyoto, Japan, October 23-27, 2022*. IEEE, 2022, p. 4112-4119. DOI : 10.1109/IROS47612.2022.9981661. URL : https://doi.org/10.1109/IROS47612.2022.9981661.

[134] Takayuki OSA, Joni PAJARINEN, Gerhard NEUMANN, J. Andrew BAGNELL, Pieter ABBEEL et Jan PETERS. « An Algorithmic Perspective on Imitation Learning ». In : *Found. Trends Robotics* 7.1-2 (2018), p. 1-179. DOI : 10.1561/2300000053. URL : https://doi.org/10.1561/2300000053.

[135] Alexandros PARASCHOS, Christian DANIEL, Jan PETERS et Gerhard NEUMANN. « Probabilistic Movement Primitives ». In : *Advances in Neural Information Processing Systems 26 : 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*. Sous la dir. de Christopher J. C. BURGES, Léon BOTTOU, Zoubin GHAHRAMANI et Kilian Q. WEINBERGER. 2013, p. 2616-2624. URL : https://proceedings.neurips.cc/paper/2013/hash/e53a0a2978c28872a4505bdb51db06dc-Abstract.html.

[136] Andreas ten PAS, Marcus GUALTIERI, Kate SAENKO et Robert Platt JR. « Grasp Pose Detection in Point Clouds ». In : *International Journal of Robotics Research* 36.13-14 (2017), p. 1455-1473. DOI : 10.1177/0278364917735594. URL : https://doi.org/10.1177/0278364917735594.

[137] Despoina PASCHALIDOU, Ali Osman ULUSOY et Andreas GEIGER. « Superquadrics Revisited : Learning 3D Shape Parsing Beyond Cuboids ». In : *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, p. 10344-10353. DOI : 10.1109/CVPR.2019.01059. URL : https://openaccess.thecvf.com/content_CVPR_2019/html/Paschalidou_Superquadrics_Revisited_Learning_3D_Shape_Parsing_Beyond_Cuboids_CVPR_2019_paper.html.

[138] Lerrel PINTO et Abhinav GUPTA. « Supersizing self-supervision : Learning to grasp from 50K tries and 700 robot hours ». In : *2016 IEEE International Conference on Robotics and Automation, ICRA 2016, Stockholm, Sweden, May 16-21, 2016*. Sous la dir. de Danica KRAGIC, Antonio BICCHI et Alessandro De LUCA. IEEE, 2016, p. 3406-3413. DOI : 10.1109/ICRA.2016.7487517. URL : https://doi.org/10.1109/ICRA.2016.7487517.

[139] Domenico PRATTICHIZZO et Jeffrey C. TRINKLE. « Grasping ». In : *Springer Handbook of Robotics*. Sous la dir. de Bruno SICILIANO et Oussama KHATIB. Springer Handbooks. Springer, 2016, p. 955-988. DOI : 10.1007/978-3-319-32552-1_38. URL : https://doi.org/10.1007/978-3-319-32552-1%5C_38.

[140] Pragathi PRAVEENA, Guru SUBRAMANI, Bilge MUTLU et Michael GLEICHER. « Characterizing Input Methods for Human-to-Robot Demonstrations ». In : *14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2019.

[141] Charles R. QI, Wei LIU, Chenxia WU, Hao SU et Leonidas J. GUIBAS. « Frustum PointNets for 3D Object Detection From RGB-D Data ». In : *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 2018, p. 918-927. DOI : 10.1109/CVPR.2018.00102. URL : http://openaccess.thecvf.com/content_cvpr_2018/html/Qi_Frustum_PointNets_for_CVPR_2018_paper.html.

[142] Charles Ruizhongtai QI, Hao SU, Kaichun MO et Leonidas J. GUIBAS. « PointNet : Deep Learning on Point Sets for 3D Classification and Segmentation ». In : *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, p. 77-85. DOI : 10.1109/CVPR.2017.16. URL : https://doi.org/10.1109/CVPR.2017.16.

[143] Charles Ruizhongtai QI, Li YI, Hao SU et Leonidas J. GUIBAS. « PointNet++ : Deep Hierarchical Feature Learning on Point Sets in a Metric Space ». In : *Advances in Neural Information Processing Systems 30 : Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Sous la dir. d'Isabelle GUYON, Ulrike von LUXBURG, Samy BENGIO, Hanna M. WALLACH, Rob FERGUS, S. V. N. VISHWANATHAN et Roman GARNETT. 2017, p. 5099-5108. URL : https://proceedings.neurips.cc/paper/2017/hash/d8bf84be3800d12f74d8b05e9b89836f-Abstract.html.

[144] Yuzhe QIN, Rui CHEN, Hao ZHU, Meng SONG, Jing XU et Hao SU. « S4G : Amodal Single-view Single-Shot SE(3) Grasp Detection in Cluttered Scenes ». In : *3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings*. Sous la dir. de Leslie Pack KAELBLING, Danica KRAGIC et Komei SUGIURA. T. 100. Proceedings of Machine Learning Research. PMLR, 2019, p. 53-65. URL : http://proceedings.mlr.press/v100/qin20a.html.

[145] Morgan QUIGLEY, Ken CONLEY, Brian GERKEY, Josh FAUST, Tully FOOTE, Jeremy LEIBS, Rob WHEELER, Andrew Y NG et al. « ROS : an open-source Robot Operating System ». In : *ICRA workshop on open source software*. 2009.

[146] LIVRE Carl Edward Rasmussen et Christopher K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

[147] Joseph Redmon et Anelia Angelova. « Real-time grasp detection using convolutional neural networks ». In : *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*. IEEE, 2015, p. 1316-1322. doi : 10.1109/ICRA.2015.7139361. url : https://doi.org/10.1109/ICRA.2015.7139361.

[148] Máximo A. Roa et Raúl Suárez. « Grasp quality measures : review and performance ». In : *Auton. Robots* 38.1 (2015), p. 65-88. doi : 10.1007/s10514-014-9402-3. url : https://doi.org/10.1007/s10514-014-9402-3.

[149] Alberto Rodriguez, Matthew T. Mason et Steve Ferry. « From caging to grasping ». In : *International Journal of Robotics Research* 31.7 (2012), p. 886-900. doi : 10.1177/0278364912442972. url : https://doi.org/10.1177/0278364912442972.

[150] Olaf Ronneberger, Philipp Fischer et Thomas Brox. « U-Net : Convolutional Networks for Biomedical Image Segmentation ». In : *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*. Sous la dir. de Nassir Navab, Joachim Hornegger, William M. Wells III et Alejandro F. Frangi. T. 9351. Lecture Notes in Computer Science. Springer, 2015, p. 234-241. doi : 10.1007/978-3-319-24574-4_28. url : https://doi.org/10.1007/978-3-319-24574-4%5C_28.

[151] Carsten Rother, Vladimir Kolmogorov et Andrew Blake. « "GrabCut" : interactive foreground extraction using iterated graph cuts ». In : *ACM Transactions on Graphics* 23.3 (2004), p. 309-314.

[152] Carlos Rubert, Daniel Kappler, Antonio Morales, Stefan Schaal et Jeannette Bohg. « On the relevance of grasp metrics for predicting grasp success ». In : *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*. IEEE, 2017, p. 265-272. doi : 10.1109/IROS.2017.8202167. url : https://doi.org/10.1109/IROS.2017.8202167.

[153] Reuven Y. Rubinstein. « Optimization of computer simulation models with rare events ». In : *European Journal of Operational Research* 99.1 (1997), p. 89-112. issn : 0377-2217. doi : https://doi.org/10.1016/S0377-2217(96)00385-2. url : https://www.sciencedirect.com/science/article/pii/S0377221796003852.

[154] Anis Sahbani, Sahar El-Khoury et Philippe Bidaud. « An overview of 3D object grasp synthesis algorithms ». In : *Robotics and Autonomous Systems* 60.3 (2012), p. 326-336. doi : 10.1016/j.robot.2011.07.016. url : https://doi.org/10.1016/j.robot.2011.07.016.

[155] Shreeyak S. Sajjan, Matthew Moore, Mike Pan, Ganesh Nagaraja, Johnny Lee, Andy Zeng et Shuran Song. « Clear Grasp : 3D Shape Estimation of Transparent Objects for Manipulation ». In : *2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020*.

IEEE, 2020, p. 3634-3642. DOI : 10.1109/ICRA40945.2020.9197518. URL : https://doi.org/10.1109/ICRA40945.2020.9197518.

[156] Jose SANCHEZ, Juan Antonio CORRALES, Belhassen-Chedli BOUZGARROU et Youcef MEZOUAR. « Robotic manipulation and sensing of deformable objects in domestic and industrial applications : a survey ». In : *International Journal of Robotics Research* 37.7 (2018), p. 688-716. DOI : 10.1177/0278364918779698. URL : https://doi.org/10.1177/0278364918779698.

[157] Vishal SATISH, Jeffrey MAHLER et Ken GOLDBERG. « On-Policy Dataset Synthesis for Learning Robot Grasping Policies Using Fully Convolutional Deep Networks ». In : *IEEE Robotics and Automation Letters* 4.2 (2019), p. 1357-1364. DOI : 10.1109/LRA.2019.2895878. URL : https://doi.org/10.1109/LRA.2019.2895878.

[158] Ashutosh SAXENA, Justin DRIEMEYER, Justin KEARNS, Chioma OSONDU et Andrew Y. NG. « Learning to Grasp Novel Objects Using Vision ». In : *Experimental Robotics, The 10th International Symposium on Experimental Robotics [ISER '06, July 6-10, 2006, Rio de Janeiro, Brazil]*. Sous la dir. d'Oussama KHATIB, Vijay KUMAR et Daniela RUS. T. 39. Springer Tracts in Advanced Robotics. Springer, 2006, p. 33-42. DOI : 10.1007/978-3-540-77457-0_4. URL : https://doi.org/10.1007/978-3-540-77457-0%5C_4.

[159] Ashutosh SAXENA, Justin DRIEMEYER et Andrew Y. NG. « Robotic Grasping of Novel Objects using Vision ». In : *International Journal of Robotics Research* 27.2 (2008), p. 157-173. DOI : 10.1177/0278364907087172. URL : https://doi.org/10.1177/0278364907087172.

[160] Stefan SCHAAL. « Is imitation learning the route to humanoid robots ? » In : *Trends in Cognitive Sciences* 3.6 (1999), p. 233-242. ISSN : 1364-6613. DOI : https://doi.org/10.1016/S1364-6613(99)01327-3. URL : https://www.sciencedirect.com/science/article/pii/S1364661399013273.

[161] Henry SCHAUB et Alfred SCHÖTTL. « 6-DOF Grasp Detection for Unknown Objects ». In : *10th International Conference on Advanced Computer Information Technologies, ACIT 2020, Deggendorf, Germany, September 16-18, 2020*. IEEE, 2020, p. 400-403. DOI : 10.1109/ACIT49673.2020.9208918. URL : https://doi.org/10.1109/ACIT49673.2020.9208918.

[162] Philipp SCHMIDT, Nikolaus VAHRENKAMP, Mirko WÄCHTER et Tamim ASFOUR. « Grasping of Unknown Objects Using Deep Convolutional Neural Networks Based on Depth Images ». In : *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*. IEEE, 2018, p. 6831-6838. DOI : 10.1109/ICRA.2018.8463204. URL : https://doi.org/10.1109/ICRA.2018.8463204.

[163] Markus SCHOELER et Florentin WÖRGÖTTER. « Bootstrapping the Semantics of Tools : Affordance Analysis of Real World Objects on a Per-part Basis ». In : *IEEE Transactions on Cognitive and Developmental Systems* 8.2 (2016), p. 84-98. DOI : 10.1109/TAMD.2015.2488284. URL : https://doi.org/10.1109/TAMD.2015.2488284.

[164] LIVRE Klaus SCHWAB. *The fourth industrial revolution*. Currency, 2017.

[165] Burr Settles. « Active learning literature survey ». In : (2009). URL : https://burrsettles.com/pub/settles.activelearning.pdf.

[166] Yu She, Shaoxiong Wang, Siyuan Dong, Neha Sunil, Alberto Rodriguez et Edward H. Adelson. « Cable manipulation with a tactile-reactive gripper ». In : *International Journal of Robotics Research* 40.12-14 (2021), p. 1385-1401. DOI : 10.1177/02783649211027233. URL : https://doi.org/10.1177/02783649211027233.

[167] Jianbo Shi et Jitendra Malik. « Normalized Cuts and Image Segmentation ». In : *1997 Conference on Computer Vision and Pattern Recognition (CVPR '97), June 17-19, 1997, San Juan, Puerto Rico*. IEEE Computer Society, 1997, p. 731-737. DOI : 10.1109/CVPR.1997.609407. URL : https://doi.org/10.1109/CVPR.1997.609407.

[168] Arjun Singh, James Sha, Karthik S. Narayan, Tudor Achim et Pieter Abbeel. « BigBIRD : A large-scale 3D database of object instances ». In : *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*. IEEE, 2014, p. 509-516. DOI : 10.1109/ICRA.2014.6906903. URL : https://doi.org/10.1109/ICRA.2014.6906903.

[169] Shuran Song, Andy Zeng, Johnny Lee et Thomas A. Funkhouser. « Grasping in the Wild : Learning 6DoF Closed-Loop Grasping From Low-Cost Demonstrations ». In : *IEEE Robotics and Automation Letters* 5.3 (2020), p. 4978-4985. DOI : 10.1109/LRA.2020.3004787. URL : https://doi.org/10.1109/LRA.2020.3004787.

[170] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang et Jan Kautz. « SPLATNet : Sparse Lattice Networks for Point Cloud Processing ». In : *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 2018, p. 2530-2539. DOI : 10.1109/CVPR.2018.00268. URL : http://openaccess.thecvf.com/content_cvpr_2018/html/Su_SPLATNet_Sparse_Lattice_CVPR_2018_paper.html.

[171] Hang Su, Subhransu Maji, Evangelos Kalogerakis et Erik G. Learned-Miller. « Multi-view Convolutional Neural Networks for 3D Shape Recognition ». In : *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*. IEEE Computer Society, 2015, p. 945-953. DOI : 10.1109/ICCV.2015.114. URL : https://doi.org/10.1109/ICCV.2015.114.

[172] Li Sun, Cheng Zhao, Zhi Yan, Pengcheng Liu, Tom Duckett et Rustam Stolkin. « A novel weakly-supervised approach for RGB-D-based nuclear waste object detection ». In : *IEEE Sensors Journal* 19.9 (2018), p. 3487-3500.

[173] Martin Sundermeyer, Arsalan Mousavian, Rudolph Triebel et Dieter Fox. « Contact-GraspNet : Efficient 6-DoF Grasp Generation in Cluttered Scenes ». In : *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*. IEEE, 2021, p. 13438-13444. DOI : 10.1109/ICRA48506.2021.9561877. URL : https://doi.org/10.1109/ICRA48506.2021.9561877.

[174] Brijen Thananjeyan, Justin Kerr, Huang Huang, Joseph E. Gonzalez et Ken Goldberg. « All You Need is LUV : Unsupervised Collection of Labeled Images using Invisible UV Fluorescent Indicators ». In : *CoRR* abs/2203.04566 (2022). DOI : 10.48550/arXiv.2203.04566. arXiv : 2203.04566. URL : https://doi.org/10.48550/arXiv.2203.04566.

[175] Yoshihisa Tsurumine, Yunduan Cui, Eiji Uchibe et Takamitsu Matsubara. « Deep reinforcement learning with smooth policy update : Application to robotic cloth manipulation ». In : *Robotics and Autonomous Systems* 112 (2019), p. 72-83. DOI : 10.1016/j.robot.2018.11.004. URL : https://doi.org/10.1016/j.robot.2018.11.004.

[176] Emre Ugur et Justus H. Piater. « Bottom-up learning of object categories, action effects and logical rules : From continuous manipulative exploration to symbolic planning ». In : *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*. IEEE, 2015, p. 2627-2633. DOI : 10.1109/ICRA.2015.7139553. URL : https://doi.org/10.1109/ICRA.2015.7139553.

[177] LIVRE Vladimir Naumovich Vapni. *The Nature of Statistical Learning Theory*. Springer, 1995. DOI : 10.1007/978-1-4757-2440-0. URL : https://doi.org/10.1007/978-1-4757-2440-0.

[178] Giulia Vezzani, Ugo Pattacini et Lorenzo Natale. « A grasping approach based on superquadric models ». In : *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*. IEEE, 2017, p. 1579-1586. DOI : 10.1109/ICRA.2017.7989187. URL : https://doi.org/10.1109/ICRA.2017.7989187.

[179] Ulrich Viereck, Andreas ten Pas, Kate Saenko et Robert Platt Jr. « Learning a visuomotor controller for real world robotic grasping using simulated depth images ». In : *Conference on Robot Learning (CoRL)*. 2017.

[180] Ruben Villegas, Arkanath Pathak, Harini Kannan, Dumitru Erhan, Quoc V. Le et Honglak Lee. « High Fidelity Video Prediction with Large Stochastic Recurrent Neural Networks ». In : *Advances in Neural Information Processing Systems 32 : Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Sous la dir. d'Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox et Roman Garnett. 2019, p. 81-91. URL : https://proceedings.neurips.cc/paper/2019/hash/f7177163c833dff4b38fc8d2872f1ec6-Abstract.html.

[181] Mohit Vohra, Ravi Prakash et Laxmidhar Behera. « Real-time Grasp Pose Estimation for Novel Objects in Densely Cluttered Environment ». In : *28th IEEE International Conference on Robot and Human Interactive Communication, RO-MAN 2019, New Delhi, India, October 14-18, 2019*. IEEE, 2019, p. 1-6. DOI : 10.1109/RO-MAN46459.2019.8956438. URL : https://doi.org/10.1109/RO-MAN46459.2019.8956438.

[182] Yongbiao WAN, Yan WANG et Chuan Fei GUO. « Recent progresses on flexible tactile sensors ». In : *Materials Today Physics* 1 (2017), p. 61-73. ISSN : 2542-5293. DOI : https://doi.org/10.1016/j.mtphys.2017.06.002. URL : https://www.sciencedirect.com/science/article/pii/S2542529317301001.

[183] Bin WANG et Piotr DUDEK. « A Fast Self-Tuning Background Subtraction Algorithm ». In : *IEEE Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2014, Columbus, OH, USA, June 23-28, 2014*. IEEE Computer Society, 2014, p. 401-404. DOI : 10.1109/CVPRW.2014.64. URL : https://doi.org/10.1109/CVPRW.2014.64.

[184] David WANG, David TSENG, Pusong LI, Yiding JIANG, Menglong GUO, Michael DANIELCZUK, Jeffrey MAHLER, Jeffrey ICHNOWSKI et Ken GOLDBERG. « Adversarial Grasp Objects ». In : *15th IEEE International Conference on Automation Science and Engineering, CASE 2019, Vancouver, BC, Canada, August 22-26, 2019*. IEEE, 2019, p. 241-248. DOI : 10.1109/COASE.2019.8843059. URL : https://doi.org/10.1109/COASE.2019.8843059.

[185] DeLiang L. WANG. « Unsupervised Learning : Foundations of Neural Computation ». In : *AI Magazine* 22.2 (2001), p. 101-102. DOI : 10.1609/aimag.v22i2.1565. URL : https://doi.org/10.1609/aimag.v22i2.1565.

[186] Dexin WANG, Chunsheng LIU, Faliang CHANG, Nanjun LI et Guangxin LI. « High-Performance Pixel-Level Grasp Detection Based on Adaptive Grasping and Grasp-Aware Network ». In : *IEEE Transactions on Industrial Electronics* 69.11 (2022), p. 11611-11621. DOI : 10.1109/TIE.2021.3120474. URL : https://doi.org/10.1109/TIE.2021.3120474.

[187] Shaoxiong WANG, Mike LAMBETA, Po-Wei CHOU et Roberto CALANDRA. « TACTO : A Fast, Flexible, and Open-Source Simulator for High-Resolution Vision-Based Tactile Sensors ». In : *IEEE Robotics and Automation Letters* 7.2 (2022), p. 3930-3937. DOI : 10.1109/LRA.2022.3146945. URL : https://doi.org/10.1109/LRA.2022.3146945.

[188] Shengfan WANG, Xin JIANG, Jie ZHAO, Xiaoman WANG, Weiguo ZHOU et Yunhui LIU. « Efficient Fully Convolution Neural Network for Generating Pixel Wise Robotic Grasps With High Resolution Images ». In : *2019 IEEE International Conference on Robotics and Biomimetics, ROBIO 2019, Dali, China, December 6-8, 2019*. IEEE, 2019, p. 474-480. DOI : 10.1109/ROBIO49542.2019.8961711. URL : https://doi.org/10.1109/ROBIO49542.2019.8961711.

[189] Yue WANG, Yongbin SUN, Ziwei LIU, Sanjay E. SARMA, Michael M. BRONSTEIN et Justin M. SOLOMON. « Dynamic Graph CNN for Learning on Point Clouds ». In : *ACM Transactions on Graphics* 38.5 (2019), 146 :1-146 :12. DOI : 10.1145/3326362. URL : https://doi.org/10.1145/3326362.

[190] Wei WEI, Yongkang LUO, Fuyu LI, Guangyun XU, Jun ZHONG, Wanyi LI et Peng WANG. « GPR : Grasp Pose Refinement Network for Cluttered Scenes ». In : *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*. IEEE, 2021, p. 4295-4302. DOI : 10.1109/ICRA48506.2021.9561868. URL : https://doi.org/10.1109/ICRA48506.2021.9561868.

[191] Ruoshi Wen, Kai Yuan, Qiang Wang, Shuai Heng et Zhibin Li. « Force-Guided High-Precision Grasping Control of Fragile and Deformable Objects Using sEMG-Based Force Prediction ». In : *IEEE Robotics and Automation Letters* 5.2 (2020), p. 2762-2769. DOI : 10.1109/LRA.2020.2974439. URL : https://doi.org/10.1109/LRA.2020.2974439.

[192] Lilian Weng. « From Autoencoder to Beta-VAE ». In : *lilianweng.github.io* (2018). URL : https://lilianweng.github.io/posts/2018-08-12-vae/.

[193] Walter Wohlkinger, Aitor Aldoma, Radu Bogdan Rusu et Markus Vincze. « 3DNet : Large-scale object class recognition from CAD models ». In : *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*. IEEE, 2012, p. 5384-5391. DOI : 10.1109/ICRA.2012.6225116. URL : https://doi.org/10.1109/ICRA.2012.6225116.

[194] Bohan Wu, Suraj Nair, Roberto Martin-Martin, Li Fei-Fei et Chelsea Finn. « Greedy Hierarchical Variational Autoencoders for Large-Scale Video Prediction ». In : *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 2021, p. 2318-2328. DOI : 10.1109/CVPR46437.2021.00235. URL : https://openaccess.thecvf.com/content/CVPR2021/html/Wu_Greedy_Hierarchical_Variational_Autoencoders_for_Large-Scale_Video_Prediction_CVPR_2021_paper.html.

[195] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang et Jianxiong Xiao. « 3D ShapeNets : A deep representation for volumetric shapes ». In : *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. IEEE Computer Society, 2015, p. 1912-1920. DOI : 10.1109/CVPR.2015.7298801. URL : https://doi.org/10.1109/CVPR.2015.7298801.

[196] Alessio Xompero, Ricardo Sanchez-Matilla, Apostolos Modas, Pascal Frossard et Andrea Cavallaro. « Multi-View Shape Estimation of Transparent Containers ». In : *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, 2020, p. 2363-2367. DOI : 10.1109/ICASSP40776.2020.9054112. URL : https://doi.org/10.1109/ICASSP40776.2020.9054112.

[197] Chi Xu, Jiale Chen, Mengyang Yao, Jun Zhou, Lijun Zhang et Yi Liu. « 6DoF Pose Estimation of Transparent Object from a Single RGB-D Image ». In : *Sensors* 20.23 (2020), p. 6790. DOI : 10.3390/s20236790. URL : https://doi.org/10.3390/s20236790.

[198] Danfei Xu, Dragomir Anguelov et Ashesh Jain. « PointFusion : Deep Sensor Fusion for 3D Bounding Box Estimation ». In : *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 2018, p. 244-253. DOI : 10.1109/CVPR.2018.00033. URL : http://openaccess.thecvf.com/content_cvpr_2018/html/Xu_PointFusion_Deep_Sensor_CVPR_2018_paper.html.

[199] Daniel YANG, Tarik TOSUN, Benjamin EISNER, Volkan ISLER et Daniel D. LEE. « Robotic Grasping through Combined Image-Based Grasp Proposal and 3D Reconstruction ». In : *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*. IEEE, 2021, p. 6350-6356. DOI : 10.1109/ICRA48506.2021.9562046. URL : https://doi.org/10.1109/ICRA48506.2021.9562046.

[200] YoungJoon YOO, Sangdoo YUN, Hyung Jin CHANG, Yiannis DEMIRIS et Jin Young CHOI. « Variational autoencoded regression : high dimensional regression of visual data on complex manifold ». In : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[201] Kuan-Ting YU et Alberto RODRIGUEZ. « Realtime State Estimation with Tactile and Visual Sensing. Application to Planar Manipulation ». In : *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*. IEEE, 2018, p. 7778-7785. DOI : 10.1109/ICRA.2018.8463183. URL : https://doi.org/10.1109/ICRA.2018.8463183.

[202] Yingying YU, Zhiqiang CAO, Shuang LIANG, Wenjie GENG et Junzhi YU. « A novel vision-based grasping method under occlusion for manipulating robotic system ». In : *IEEE Sensors Journal* 20.18 (2020), p. 10996-11006.

[203] Wenzhen YUAN, Siyuan DONG et Edward H. ADELSON. « GelSight : High-Resolution Robot Tactile Sensors for Estimating Geometry and Force ». In : *Sensors* 17.12 (2017), p. 2762. DOI : 10.3390/s17122762. URL : https://doi.org/10.3390/s17122762.

[204] Andy ZENG, Shuran SONG, Stefan WELKER, Johnny LEE, Alberto RODRIGUEZ et Thomas A. FUNKHOUSER. « Learning Synergies Between Pushing and Grasping with Self-Supervised Deep Reinforcement Learning ». In : *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018*. IEEE, 2018, p. 4238-4245. DOI : 10.1109/IROS.2018.8593986. URL : https://doi.org/10.1109/IROS.2018.8593986.

[205] Andy ZENG, Shuran SONG, Kuan-Ting YU, Elliott DONLON, Francois Robert HOGAN, Maria BAUZÁ, Daolin MA, Orion TAYLOR, Melody LIU, Eudald ROMO, Nima FAZELI, Ferran ALET, Nikhil Chavan DAFLE, Rachel M. HOLLADAY, Isabella MORONA, Prem Qu NAIR, Druck GREEN, Ian H. TAYLOR, Weber LIU, Thomas A. FUNKHOUSER et Alberto RODRIGUEZ. « Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching ». In : *International Journal of Robotics Research* 41.7 (2022), p. 690-705. DOI : 10.1177/0278364919868017. URL : https://doi.org/10.1177/0278364919868017.

[206] Hanbo ZHANG, Jian TANG, Shiguang SUN et Xuguang LAN. « Robotic Grasping from Classical to Modern : A Survey ». In : *CoRR* abs/2202.03631 (2022). arXiv : 2202.03631. URL : https://arxiv.org/abs/2202.03631.

[207] Hao ZHANG, Hongzhuo LIANG, Lin CONG, Jianzhi LYU, Long ZENG, Pingfa FENG et Jianwei ZHANG. « Reinforcement Learning Based Pushing and Grasping Objects from Ungraspable Poses ». In : *CoRR* abs/2302.13328 (2023). DOI : 10.48550/arXiv.2302.13328. arXiv : 2302.13328. URL : https://doi.org/10.48550/arXiv.2302.13328.

[208] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg et Pieter Abbeel. « Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation ». In : *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*. IEEE, 2018, p. 1-8. DOI : 10.1109/ICRA.2018.8461249. URL : https://doi.org/10.1109/ICRA.2018.8461249.

[209] Binglei Zhao, Hanbo Zhang, Xuguang Lan, Haoyu Wang, Zhiqiang Tian et Nanning Zheng. « Regnet : Region-based grasp network for end-to-end grasp detection in point clouds ». In : *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, p. 13474-13480.

[210] Xupeng Zhu, Dian Wang, Ondrej Biza, Guanang Su, Robin Walters et Robert Platt. « Sample Efficient Grasp Learning Using Equivariant Models ». In : *Proceedings of Robotics : Science and Systems (RSS)* (2022).

[211] Xupeng Zhu, Dian Wang, Ondrej Biza, Guanang Su, Robin Walters et Robert Platt. « Sample Efficient Grasp Learning Using Equivariant Models ». In : *CoRR* abs/2202.09468 (2022). arXiv : 2202.09468. URL : https://arxiv.org/abs/2202.09468.

[212] Zoran Zivkovic. « Improved Adaptive Gaussian Mixture Model for Background Subtraction ». In : *17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23-26, 2004*. IEEE Computer Society, 2004, p. 28-31. DOI : 10.1109/ICPR.2004.1333992. URL : https://doi.org/10.1109/ICPR.2004.1333992.

[213] Zoran Zivkovic et Ferdinand van der Heijden. « Efficient adaptive density estimation per image pixel for the task of background subtraction ». In : *Pattern Recognition Letters* 27.7 (2006), p. 773-780. DOI : 10.1016/j.patrec.2005.11.005. URL : https://doi.org/10.1016/j.patrec.2005.11.005.