Whole-body Teleoperation of Humanoid Robots

Luigi Penco

▶ **To cite this version:**

Luigi Penco. Whole-body Teleoperation of Humanoid Robots. Robotics [cs.RO]. Université de Lorraine, 2022. English. NNT : 2022LORR0059 . tel-03751078

HAL Id: tel-03751078
https://inria.hal.science/tel-03751078

Submitted on 13 Aug 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Whole-body Teleoperation of Humanoid Robots

THÈSE

présentée et soutenue publiquement le 7 juin 2022

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Luigi Penco

Composition du jury

Rapporteurs :	Dongheui Lee	Professeur, TU Wien
	Olivier Stasse	Directeur de Recherche CNRS, LAAS Toulouse
Examinateurs :	Paolo Robuffo Giordano	Directeur de Recherche CNRS, Inria/IRISA Rennes
	Jerry Pratt	Directeur de Recherche, Florida IHMC
Encadrants :	Jean-Baptiste Mouret	Directeur de Recherche CNRS, Inria Nancy Grand-Est
	Serena Ivaldi	Chargée de Recherche CNRS, Inria Nancy Grand-Est

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

... ἔοικα γοῦν τούτου γε σμικρῷ τινι αὐτῷ τούτῳ σοφώτερος εἶναι,
ὅτι ἃ μὴ οἶδα οὐδὲ οἴομαι εἰδέναι.

(in Plato, Apology 21d)

... I seem, then, in just this little thing to be wiser than this man at any rate,
that what I do not know I do not think I know either.

(from the Henry Cary literal translation of 1897)

Contents

List of Figures

List of Tables

Summary in French

Tout au long de l'histoire, l'homme a souvent été le principal sujet des arts visuels, car de nombreuses œuvres d'art n'avaient d'autre but explicite que de produire de la beauté. Les œuvres d'art devaient être regardées et admirées et susciter des émotions, et seuls les sujets humains pouvaient transmettre certains types d'émotions. L'idée de créer des robots semblables aux humains a d'abord été conçue (et l'est encore souvent aujourd'hui) pour satisfaire cet esprit artistique, ainsi que le désir humain de recréer et éventuellement d'améliorer son propre corps. Le concept de robots humanoïdes est né dans de nombreuses cultures anciennes du monde entier, et a fait son apparition dans la mythologie grecque au début du IVe siècle avant Jésus-Christ. L'un des premiers dessins enregistrés d'un robot humanoïde a été réalisé par Léonard de Vinci vers 1495. Ses carnets, redécouverts dans les années 1950, contiennent des dessins détaillés d'un chevalier mécanique en armure (Figure 1 gauche) qui était censé s'asseoir, agiter les bras et bouger la tête et la mâchoire. Les preuves de l'existence de robots humanoïdes construits n'apparaissent que trois siècles plus tard. Le premier robot humanoïde construit de l'histoire serait en fait un soldat équipé d'un soufflet automatique soufflant dans une trompette (figure 1 droite), fabriqué en 1810 par Friedrich Kaufmann à Dresde, en Allemagne.

En 1928, un robot humanoïde plus avancé a été présenté à l'exposition annuelle de la Model Engineers Society de Londres. Inventé par W. H. Richards, le robot était constitué d'une armure en aluminium avec onze électro-aimants et un moteur alimenté par une source de courant de 12 volts. Le robot pouvait bouger ses mains et sa tête et pouvait être contrôlé par télécommande. En 1962, la première société de robotique au monde, Unimation, a été fondée. Elle a ensuite posé les bases de l'industrie robotique moderne. En 2000, Honda a révélé le développement d'un nouveau robot humanoïde léger et de petite taille, ASIMO, qui utilise la nouvelle technologie de marche robotique de Honda pour atteindre une capacité de marche semblable à celle d'un humain. En 2013, Boston Dynamics a lancé le robot humanoïde Atlas [2], qui est aujourd'hui le plus avancé en termes de capacités de locomotion. Il peut en effet traverser des terrains accidentés, exécuter quelque chose qui s'apparente à une routine au sol en gymnastique, et effectuer un circuit de parkour avec des sauts et des poutres d'équilibre. La quantité de robots humanoïdes a maintenant augmenté au point que plus de 70 plateformes humanoïdes différentes existent dans le monde, dont certaines sont également disponibles sur le marché.

Les humanoïdes sont utilisés comme outils de recherche dans plusieurs domaines scientifiques et sont développés dans le but d'être des travailleurs mécaniques polyvalents ou des platformes de divertissement. En raison de leur complexité, leurs capacités sont encore limitées et ne sont pas comparables au niveau de compétences et de polyvalence des humains. Leur utilité est parfois remise en question au sein de la communauté robotique, car il existe des situations où des robots plus simples peuvent résoudre un problème plus efficacement. Un lave-linge, par exemple, est une machine simple, très efficace pour laver le linge et qui présente peu de similitudes avec la façon dont l'homme le fait. De même, il existe des situations où des robots mobiles comme des rovers ou de simples robots manipulateurs suffisent pour accomplir une certaine tâche. Cependant, la poly-

valence potentielle des robots humanoïdes en fait des plateformes intéressantes dans l'optique d'une société future où les robots sont censés être bien intégrés dans la vie quotidienne des humains. En effet, les gens vivent dans des environnements qui s'adaptent à leur morphologie et à leur comportement. Par exemple, les objets du quotidien sont conçus pour tenir dans la main d'une personne et sont suffisamment légers pour être transportés ; les tables et les chaises sont conçues pour avoir une hauteur adaptée au corps humain ; les escaliers sont conçus pour relier des zones à différentes hauteurs, d'une manière adaptée à la locomotion humaine. Les robots humanoïdes peuvent potentiellement tirer parti de ces conceptions centrées sur l'homme, ce qui simplifie les tâches et évite de devoir modifier l'environnement pour le robot. De plus, le contexte dans lequel les robots sont censés opérer est caractérisé par un haut niveau d'incertitude dynamique, et par la présence de plusieurs collaborateurs humains ou spectateurs. Dans ces circonstances, le robot doit être capable d'interagir socialement et physiquement avec ses homologues humains, pour lesquels une morphologie humaine aide réellement à rendre les actions du robot "lisibles" et "prévisibles" par les partenaires humains [53].

Enfin, alors que les chercheurs étudient la structure et le comportement du corps humain pour construire des robots humanoïdes, la même tentative d'imiter le corps humain conduit à une meilleure compréhension de celui-ci. Cela a par exemple permis de mettre au point des prothèses de jambe ou d'avant-bras motorisées pour les personnes souffrant de troubles neuromusculaires ou de réaliser de nombreuses études d'ergonomie sur le mouvement humain, qui peuvent contribuer à réduire le risque de blessure dans les industries [**?**].

Étant hautement redondants et dotés d'une structure corporelle complexe, les robots humanoïdes sont difficiles à contrôler, en particulier dans les applications du monde réel impliquant la locomotion, ainsi que l'interaction avec l'environnement [102] et avec des partenaires humains [158]. L'incertitude dynamique de ces environnements présente des défis critiques pour les systèmes de cognition des robots, qui, malgré les progrès récents basés sur les techniques d'apprentissage automatique, ne peuvent pas produire de manière autonome des comportements de robot socialement et physiquement compétents dans toutes les situations.

Une alternative à l'adoption de solutions entièrement autonomes peu fiables consiste à contrôler à distance le robot comme un avatar qui reproduit en temps réel le comportement humain. De cette façon, les excellentes capacités cognitives et les décisions de haut niveau de l'homme sont pleinement exploitées et transférées dans le robot. Cette technique est communément appelée "téléopération". L'un des principaux avantages de la téléopération est de permettre le contrôle à distance de robots dans des scénarios de réponse aux catastrophes ou sur des lieux de travail dangereux, où la présence humaine serait plutôt évitée.

Le terme téléopération en général, indique l' opération d'un système (le robot) à distance, et a une signification similaire au terme contrôle à distance. Par conséquent, notre interprétation précédente, à savoir "opération consistant à transférer des mouvements humains en temps réel au robot en imitant l'opérateur", est certainement valable. Cependant, ce n'est pas la définition la plus générale puisque la téléopération d'un robot peut se faire avec d'autres entrées qui ne sont pas le mouvement humain, par exemple le bouton d'un joystick.

La téléopération a été conçue pour la première fois au laboratoire national d'Argonne, peu après la Seconde Guerre mondiale, pour manipuler des matériaux radioactifs à l'aide de robots manipulateurs [67]. Par la suite, son utilisation s'est étendue à de multiples domaines, attirant l'attention de la communauté des chercheurs. En 1980, Susumu Tachi a proposé l'idée de la télexistence [176], un concept d'autonomisation humaine permettant aux humains d'exister virtuellement dans un autre lieu où ils pourraient agir librement en vue d'une société plus écologique et plus efficace en termes de temps, avec un meilleur équilibre global entre vie professionnelle et vie privée. Ce concept a été développé à travers plusieurs projets de recherche et de développement

qui ont abouti à la création d'une entreprise (Telexistence Inc.).

Au fil des années, l'intérêt pour les applications potentielles de la télexistence, et plus généralement de la téléopération, dans une multitude de domaines industriels, a augmenté. Dans le cadre de ce processus d'industrialisation de la téléopération robotique et inspirée par une visite au laboratoire Tachi, la Fondation XPRIZE a récemment lancé l'ANA Avatar XPRIZE, un concours mondial de quatre ans qui débutera en janvier 2020, axé sur le développement d'un système d'avatar qui "déploiera les sens, les actions et la présence d'un humain dans un lieu distant en temps réel, ce qui conduira à un monde plus connecté."

Aujourd'hui, la téléopération des robots est également largement employée dans les applications spatiales. Des expériences ont déjà été menées avec des robots téléopérés à bord de la station spatiale internationale (ISS), dans le cadre des projets METERON et Kontur-2 [107]. Ces projets se sont limités à l'utilisation de rovers et de manipulateurs mobiles, mais au cours de la dernière décennie, on a commencé à étudier la possibilité d'employer des robots humanoïdes pour l'exploration spatiale. Dans cette direction, l'humanoïde Robonaut 2 a été envoyé à la Station spatiale internationale (ISS) en 2011 pour aider les astronautes dans diverses tâches [52] mais a cessé de fonctionner en 2014 en raison d'un court-circuit induit par l'ajout de nouvelles jambes, et a finalement été renvoyé sur Terre après 3 ans. De même, le robot Skybot F-850 a été envoyé par fusée vers l'ISS en août 2019 dans un vaisseau spatial non habité transportant des fournitures [11]. Le robot s'est avéré avoir un design qui ne fonctionne pas bien, démontrant qu'il y a encore du travail à faire pour avoir des humanoïdes dans l'espace.

Des efforts de recherche supplémentaires sont nécessaires pour contrôler avec succès ce type de robots dans des scénarios réels, mais ils restent les candidats idéaux pour remplacer les humains dans des environnements dangereux, éloignés ou inaccessibles. En fait, contrairement aux plates-formes robotiques plus conventionnelles, la structure des humanoïdes est mieux adaptée aux environnements et aux tâches conçus pour et exécutés par des humains (voir la figure 4) et leur polyvalence opérationnelle les rend aptes à des activités professionnelles qui nécessitent une variété de mouvements complexes, comme l'inspection, la maintenance et l'interaction avec les humains. Dans certains contextes, comme la télexistence, où l'on s'attend à ce que d'autres partenaires humains et/ou des spectateurs interagissent avec le robot téléopéré, le facteur de ressemblance avec l'humain est encore plus important puisqu'il augmente l'acceptabilité et l'engagement envers le robot [53].

De nombreux efforts de la communauté robotique ont été consacrés à l'amélioration de la téléopération des humanoïdes et de nombreux défis doivent encore être relevés. Le travail de cette thèse vise à développer un système de téléopération pour les robots humanoïdes qui résout certains de ces défis.

Dans le chapitre 1, nous donnons d'abord un aperçu de l'architecture générale partagée par les systèmes de téléopération existants. Tous ces systèmes comprennent généralement la perception des états humains, leur mise en correspondance avec le robot (également appelée *motion retargeting*), un contrôleur de robot et la perception des états du robot et de l'environnement distant qui représente le retour d'information pour l'opérateur. Nous présentons les différents dispositifs de mesure qui peuvent être utilisés pour capturer les mouvements, les commandes et/ou les états de l'homme, ainsi que les interfaces de retour possibles qui permettent à l'homme de percevoir les états du robot. Dans notre implémentation, nous nous sommes appuyés sur un système de capture de mouvement basé sur la technologie inertielle pour suivre les mouvements humains et avons utilisé un casque de réalité virtuelle pour transmettre à l'utilisateur la vue des caméras du robot et d'autres caméras externes.

Dans le chapitre 2, nous présentons notre stratégie de reciblage. Nous avons suivi le mouvement humain avec la combinaison de motion capture et nous avons traité les informations hu-

maines pour produire des comportements équivalents sur le robot en temps réel. Le système de capture de mouvement produit un squelette humain reconstruit. À partir de celui-ci, nous avons converti les angles des articulations en valeurs correspondantes pour les robots iCub et HRP-4C. Nous avons mis à l'échelle les positions cartésiennes relatives et utilisé une carte d'identité entre le mouvement de rotation de l'humain et du robot. Nous avons également reciblé les trajectoires du centre de masse de l'humain au robot en recourant à des décalages normalisés qui nous ont permis de reconstruire la position du centre de masse du robot sur le sol. Enfin, nous avons proposé une correction dynamique du centre de masse : si l'utilisateur effectue un mouvement qui peut déséquilibrer le robot, son mouvement est corrigé avant d'être transféré au robot afin de garantir l'équilibre du robot. D'après les différentes configurations que nous avons testées, il semble que le suivi de la posture du corps entier de l'utilisateur, ainsi que la pose des effecteurs terminaux et la position du centre de masse au sol, produisent le comportement du robot le plus proche de l'homme. Ceci est possible si les contraintes dynamiques comme les contraintes du cône de friction sont gérées par le contrôleur. Pour les mouvements très dynamiques, une correction dynamique des références reciblées peut également être nécessaire. Nous avons proposé ici une correction simple du centre de masse basée sur LIP pour les mouvements à double appui. Des modèles simplifiés étendus devraient être utilisés pour des mouvements plus complexes impliquant le changement de contacts.

Dans le chapitre 3, nous passons en revue les contrôleurs de corps entier qui ont été utilisés dans la littérature pour téléopérer des robots humanoïdes. Un choix courant consiste à formuler le problème de commande sous la forme d'un programme QP, qui peut être utilisé pour calculer les commandes des robots commandés en position ou en couple. Dans le premier cas, le problème QP représente un problème de dynamique inverse ou de cinématique inverse, tandis que dans le second, il représente un problème de contrôle basé sur la quantité de mouvement. Nous avons utilisé un contrôleur QP basé sur IK pour le robot iCub et un QP basé sur TSID pour un robot HRP-4C simulé. Chaque contrôleur est caractérisé par plusieurs paramètres qui dictent la façon dont les trajectoires de référence des tâches sont suivies, et établissent la priorité assignée à chaque tâche. Le réglage des paramètres de contrôle nécessite normalement une procédure d'essais et d'erreurs qui ne produit pas nécessairement des comportements efficaces du robot. Pour cette raison, nous avons proposé un cadre pour apprendre automatiquement à la fois la structure et les paramètres d'un contrôleur "générique" du corps entier. Nous avons utilisé l'optimisation multi-objectifs pour rechercher des solutions sur le front de Pareto, représentant les compromis optimaux de performance (suivi des trajectoires de référence) et de robustesse (limitation du moment de basculement du robot). Cela nous a permis de trouver efficacement un ensemble de solutions de compromis Pareto-optimales en simulation. L'utilisateur peut facilement choisir dans le front de Pareto une solution pré-optimisée à tester sur le robot réel : cela réduit considérablement le nombre d'essais sur le robot réel et améliore le réglage des paramètres. Pour donner un exemple au lecteur, il ne nous a fallu que trois essais sur le robot réel pour trouver une configuration de contrôleur adaptée à la téléopération d'un humanoïde. Avant notre méthode, le réglage d'un tel contrôleur était une procédure d'essais et d'erreurs qui prenait beaucoup de temps. Dans notre système nous avons optimisé un contrôleur QP basé sur IK que nous utilisions pour téléopérer le robot iCub tout en effectuant des mouvements de double appui. Les critères de robustesse à prendre en compte pour des mouvements plus dynamiques, comme la marche ou la course, restent une question ouverte et doivent être étudiés plus en profondeur.

Dans le chapitre 4, nous avons amélioré le système de téléopération en intégrant un mode de niveau supérieur pour contrôler le robot pendant la locomotion. Grâce à cette intégration, l'utilisateur a pu passer d'un mode de téléopération de bas niveau, dans lequel il peut générer des mouvements du corps entier pour le robot grâce à une combinaison de motion capture, à un mode

de haut niveau, dans lequel il peut utiliser un joystick pour envoyer des commandes de référence au robot, telles que la direction et la vitesse du mouvement, sans s'occuper de leur exécution réelle. Dans les deux cas, l'opérateur humain reçoit un retour visuel grâce à un casque de réalité virtuelle relié aux caméras du robot. Dans le mode de téléopération de haut niveau, nous avons utilisé un générateur de démarche MPC basé sur un modèle LIP. Pour la locomotion 3D, des modèles plus complexes devraient être utilisés et, selon leur nature, une formulation non linéaire du MPC pourrait être requise.

Enfin, dans le chapitre 5, nous avons abordé les principaux problèmes qui apparaissent lors de la téléopération de robots dans des réseaux de communication non idéaux. Les retards dans la transmission des commandes de l'utilisateur au robot et dans la réception du retour d'information du robot à l'utilisateur, peuvent perturber irrémédiablement l'opérateur. Nous avons donc proposé une approche de compensation des délais qui permet de synchroniser les mouvements de l'utilisateur avec le retour visuel du robot. Le robot anticipe le mouvement de l'homme de sorte que le retour "anticipé" retardé soit synchronisé avec les commandes de l'opérateur. Dans notre implémentation, nous avons utilisé les ProMPs pour représenter les primitives du mouvement humain et aussi pour prédire les intentions de l'opérateur. Nous avons montré que le robot suit l'exécution spécifique du mouvement, c'est-à-dire la façon particulière dont l'humain effectue la tâche, en mettant continuellement à jour la prédiction du mouvement actuel (grâce à l'opérateur de conditionnement des ProMPs). Dans les expériences, nous avons pu atteindre une bouteille d'une manière qui n'avait pas été démontrée auparavant (mais incluse dans la distribution des démonstrations d'entraînement), en évitant de nouveaux obstacles et en atteignant de nouvelles positions d'objets qui n'avaient pas été incluses pendant la phase d'entraînement. Cette fonctionnalité est extrêmement précieuse car elle permet à l'opérateur d'adapter les commandes à la volée à des situations différentes de celles de la formation. Par exemple, il peut y avoir un obstacle qui oblige l'opérateur à s'approcher d'un objet avec la main droite et l'empêche de suivre un chemin droit. Ou encore, il peut y avoir un plafond bas qui oblige l'opérateur à plier davantage le torse ou à fléchir davantage les jambes. Plus important encore, il existe de nombreux objets différents et de nombreux environnements différents, et l'opérateur adaptera son mouvement aux conditions locales, d'une position cible différente à une posture corporelle différente. Grâce à un ensemble diversifié de ProMPs et à la "conformation" de la prédiction aux observations, notre système devrait être capable de gérer la plupart de ces cas. La principale limitation est que notre approche suppose que les données reçues sont suffisantes pour mettre à jour les prédictions ; c'est souvent le cas, car les humains ont tendance à anticiper leur changement dans une trajectoire, c'est-à-dire que le début de la trajectoire change lorsqu'un objectif futur change. Néanmoins, le système n'est pas capable d'anticiper un changement de dernière seconde lorsque le début de la trajectoire est toujours le même. Les travaux futurs devraient évaluer combien de démonstrations (et combien de ProMPS) sont nécessaires pour s'adapter à un maximum de situations.

Le système que nous avons développé nous a permis d'effectuer des tâches simples comme ramasser des boîtes, atteindre des objets, ouvrir des portes et imiter l'humain en espace libre. Ce qui nous a empêché d'aborder des scénarios de téléopération plus compliqués était principalement la fragilité des mains de l'iCub. Malgré leurs nombreux degrés de liberté, elles ne peuvent pas soulever des objets d'un poids supérieur à 500g. De même, le robot simulé HRP-4C présente une dextérité limitée dans les bras en raison du nombre restreint de degrés de liberté des mains et des poignets (seulement deux chacun). Des mains plus complexes et plus robustes nécessiteraient toutefois davantage d'informations sur l'état du robot et de son environnement.

En particulier, le retour haptique peut fournir des informations extrêmement précieuses pendant la manipulation. Alors qu'une interaction physique sûre avec l'environnement peut être assurée par le contrôle de bas niveau du robot sans retour haptique, ces informations sont essentielles

pour améliorer les capacités de contrôle à distance de l'opérateur [20, 154]. En général, le retour haptique est localisé dans les effecteurs terminaux, où la plupart des interactions se produisent ; dans le cas des humanoïdes téléopérés, le retour haptique sur l'ensemble du corps devrait être envisagé [149], éventuellement au moyen de dispositifs vibro-tactiles portables [32].

Afin d'effectuer des mouvements plus dynamiques, les forces au niveau des contacts doivent être contrôlées, c'est-à-dire qu'il faut revenir au contrôle du couple. Par rapport au contrôle de la position, cela offre un certain nombre d'avantages, notamment pour les robots humanoïdes. Le contrôle du couple permet d'améliorer les performances de suivi et des gains de rétroaction plus faibles peuvent être utilisés avec le contrôleur, ce qui entraîne une meilleure conformité du système. Cette conformité plus élevée permet de s'adapter automatiquement à un environnement incertain (par exemple, marcher sur un terrain accidenté) et de sécuriser l'interaction homme-robot. Cependant, l'obtention de comportements robotiques fiables avec le contrôle du couple représente toujours un défi pour la majorité des plateformes robotiques existantes. En effet, alors qu'en simulation le contrôle du couple articulaire peut être presque parfait, sur le robot réel, il y a des erreurs dans le modèle de la chaîne d'actionnement, du bruit dans les capteurs, une largeur de bande de couple limitée et/ou des retards [153]. Par conséquent, de nombreux comportements obtenus en simulation ne peuvent pas être transférés facilement sur le robot réel.

Une autre amélioration future du système que nous avons proposé dans cette thèse, pourrait être liée à l'utilisation d'approches de contrôle partagé [148, 166]. Celles-ci permettent de combiner les capacités du robot et de l'opérateur pour réaliser des tâches complexes plus efficacement. L'opérateur guide le robot grâce à ses décisions de haut niveau et à ses capacités cognitives avancées, tandis que le robot exécute les tâches de manière autonome ou semi-autonome en exploitant ses capacités de précision qui peuvent souvent surpasser celles de l'opérateur humain. Par exemple, dans le cas de [148], le robot modifie l'orientation de ses effecteurs pour aider l'opérateur à accomplir des tâches bi-manuelles plus efficacement. Une alternative intéressante qui a été récemment proposée est de guider l'exécution d'une tâche par l'opérateur par le biais d'un retour haptique [114]. Dans ce cas, le retour haptique ne représente plus les forces réelles subies du côté du robot mais il est utilisé pour guider le mouvement humain. Le niveau de guidage robotique est modulé en fonction de la prédiction des intentions de mouvement de l'homme.

Dans toutes ces approches, la téléopération devient une instance de collaboration homme-robot. Cette collaboration n'est pas unilatérale (c'est-à-dire que l'opérateur contrôle le robot qui suit passivement les commandes de l'utilisateur), mais les deux parties s'entraident pour accomplir une tâche de manière plus fluide et efficace. En ce sens, notre approche de compensation des retards peut également être considérée comme une stratégie de collaboration. Tant que l'opérateur humain est celui qui prend les décisions de haut niveau pour le robot et n'est pas privé de la possibilité d'en prendre le contrôle total, toute assistance du robot ne peut être que bénéfique. En fait, nous supposons que si l'opérateur devait fournir trop d'efforts pour contrôler le robot lors de l'accomplissement de tâches faciles, l'expérience d'incarnation serait immédiatement rompue, rendant l'opérateur frustré ou perdant patience. Il s'agira d'une caractéristique essentielle des futures applications de télexistence.

Introduction

Throughout history, humans have often been the main subject of visual arts, because much artwork was made for no other explicit purpose than the production of beauty. Artwork was to be beheld and admired and make people emotional, and only human subjects could transmit certain types of emotions. The idea of creating human-alike robots was first (and often today still is) conceived to satisfy this artistic spirit, along with a human desire to recreate and possibly improve their own body. The concept of humanoid robots originated from many different ancient cultures around the world, and was first found in the Greek mythology in the early 4-th century BC. One of the first recorded designs of a humanoid robot was made by Leonardo da Vinci in around 1 495. His notebooks, rediscovered in the 1950s, contain detailed drawings of a mechanical knight in armour (Figure 1 left) which was supposed to sit up, wave its arms and move its head and jaw. Evidence of built humanoid robots only surfaces three centuries later. The first built humanoid robot in history is in fact said to be a soldier with automatic bellows blowing a trumpet (Figure 1 right), made in 1810 by Friedrich Kaufmann in Dresden, Germany. In 1928, a more advanced humanoid robot was exhibited at the annual exhibition of the Model Engineers Society in London. Invented by W. H. Richards, the robot consisted of an aluminium suit of armour with eleven electromagnets and one motor powered by a 12-volt power source. The robot could move its hands and head and could be controlled by remote control. In 1962, the world's first robotics company Unimation was founded. This later laid the foundations of the modern robotics industry. In 1970, four laboratories in Waseda University's School of Science and Engineering teamed up and started the Wabot project. The efforts were led by Professor Ichiro Kato, a pioneer in humanoid robotics. In 1973, the group unveiled the Wabot 1. It was the world's first full-scale anthropomorphic robot, capable of walking with a quasi-dynamic gait. In 2000, Honda revealed the development of a new small, lightweight humanoid robot named ASIMO [1] that employed Honda's new robotic walking technology to achieve an unprecedented human-like ability to walk. In 2013, Boston Dynamics launched the humanoid robot Atlas [2], which is today the most advanced in terms of locomotion capabilities. It can traverse uneven terrains, perform something akin to a floor routine in gymnastics, and run a parkour course with jumps, balance beams, and vaults. The number of humanoid robots has now increased to the point that more than 70 different humanoid platforms exist around the world, some of which are also available on the market.

Humanoids are used as research tools in several scientific areas and are developed to serve as general-purpose mechanical workers or entertainers. Due to their complexity, their capabilities are still limited and not comparable to the level of skills and versatility of the humans. Their usability is at times questioned within the robotics community, since there are situations where simpler robots can resolve a problem more efficiently. A washing machine, for example, is a simple machine, very effective in washing clothes and it shares little similarity with the human way of doing it. Similarly, there are situations where mobile-based robots like rovers or simple robot manipulators are enough to achieve a certain task. However, the potential versatility that humanoid

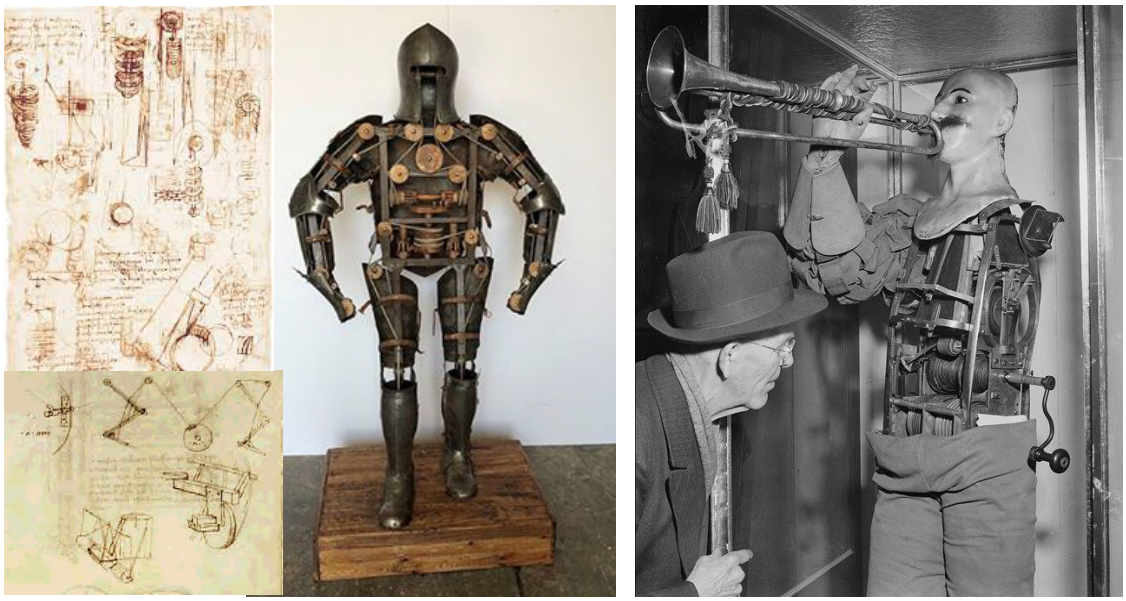


Figure 1: First evidence of the realization of the concept of humanoid robot. Left: First recorded designs of humanoid robots by Leonardo da Vinci (1495) and reconstruction of the Leonardo's robot from the original drawings (credits: lucasliso i.pinimg.com). Right: The supposedly first built humanoid robot is a soldier with a trumpet made by Friedrich Kaufmann (1810) (credits: AP Photo/Heinrich Sanden).

robots have, still makes them interesting platforms in the view of a future society where robots are supposed to be well-integrated in the daily life of humans. In fact, people inhabit environments that accommodate human morphology and human behavior. For example, everyday objects are designed to fit in a person's hands and are light enough to be carried around; tables and chairs are designed to have a height convenient for the human body; stairs are designed to connect areas at different heights, in a way that is adapted to the human locomotion. Humanoid robots can potentially take advantage of these human-centered designs, thereby simplifying tasks and avoiding the need to alter the environment for the robot. Moreover, the context in which the robots are expected to operate is characterized by a high level of dynamic uncertainty, and by the presence of several human collaborators or bystanders. In these circumstances, the robot must be able to socially and physically interact with the human counterparts, for which a human morphology really helps making the robot actions "readable", "legible" and "predictable" by the human partners [53]. Finally, while researchers study the human body structure and behavior to build humanoid robots, the same attempt to imitate the human body is leading to a better understanding of it. This for example, helped develop powered leg or forearm prosthesis for neuromuscularly impaired people or led to numerous ergonomics studies on the human motion, which can help reducing the risk of injury in the industries [84].

Being highly redundant and having a complex body structure, humanoid robots are difficult to control, particularly in real-world applications involving locomotion, as well as interaction with the environment [102] and with human partners [158]. The dynamic uncertainty of these environments presents critical challenges to the robots' cognition systems, which despite the recent progress based on machine-learning techniques, cannot autonomously produce socially and physically competent robot behaviors in all situations.

An alternative to adopting unreliable fully autonomous solutions is to remotely control the

Figure 2: Examples of hazardous working environments where robotic avatars could replace humans. Left: Asbestos tiles removal operation (credits: myjobquote.co.uk). Center: Maintenance operation in a confined space for oil and gas (credits: blog.rmiwyoming.com). Right: Health-care workers assisting contagious patients (credits: aa.com.tr).

robot as an avatar that replicates in real-time the human behavior. In this way, the excellent human cognition skills and high-level decisions are fully exploited and transferred into the robot. This technique is commonly referred to as *teleoperation*. One of the main benefits of teleoperation is enabling the remote control of robots in disaster response scenarios or dangerous workplaces, where the human presence would rather be avoided.

Studies [128], [78], [178] indicate that hundreds of thousands of workers die each year worldwide in the workplace, with a staggering cost (around 4% of global GDP) due to the time loss, worker compensation, interruption of production, and medical expenses. Not surprisingly, data show that some work activities remain inherently dangerous even with strict work regulations in place. A study [178] from 2012 identifies cancer, respiratory disease, and accidents as the major causes of work-related deaths that could be prevented through workplace automation. In particular, a survey of the Bureau of Labor Statistics [4] shows that the most common accidents in the US are fatal falls, collisions with objects and equipment and injuries in confined spaces. The number of these casualties could be reduced by avoiding the physical presence of the operator. For example, removing asbestos roof tiles is an operation that could be performed by teleoperated robots. Currently this task is carried out by humans in a context that is extremely dangerous for their health, not only because they have to move on roofs but also because they are exposed to asbestos particles (Figure 2, left). In the oil and gas industry, workers are often required to enter confined spaces for inspection and maintenance, exposing themselves to hazards such as toxic vapors, not to mention the difficulties of evacuation in case of accidents (Figure 2, center). In the hospitals, healthcare workers are exposed to infectious diseases and are at higher risk of infection compared with the general community. This is evident during disease outbreaks, as experienced during the recent COVID-19 pandemic. Nurses physically interact with the environment for a wide variety of tasks such as handling contaminated objects, taking measurements, and interact with patients (Figure 2, right). In this respect, robots teleoperated by operators could facilitate the situation and improve the nurses' safety by performing some of their tasks, as an example in [105].

Even more human lives are put at risk in disaster response scenarios, where at times more rescuers can die than the original victims. The Occupational Safety and Health Administration recently examined reports for fatal, confined-space accidents and found that when multiple deaths occurred, the majority of the victims were rescuers [10]. In certain disasters, responders cannot even be sent to the site. For example, back in March 2011, no responder could be sent in the Fukushima Daiichi power plant, because of the deadly levels of radiations. The nuclear disaster could have been vastly mitigated if a teleoperated robot entered the facility within the first 24 hours after the cooling system malfunctioned, stabilizing the first nuclear reactor. Some robots

were actually sent but they ended up being of no help since they were not able to access certain key areas by climbing stairs and opening doors, and could not perform important tasks like closing some valves, proving that mobile-based manipulator have limited capabilities and humanoid robot are needed. In all the above-mentioned scenarios human lives are exposed to serious risks and employing robots at the place of humans could be extremely valuable, relieving humans from any potential hazard.

The benefits of teleoperation are not limited to real-time applications. In fact, teleoperation can be used to generate robot motions more easily. Current approaches for motion generation in humanoid robots essentially rely on planners that search for the optimal sequence of joint commands (typically joint torques or velocities) according to an objective function that fulfills multiple tasks under several constraints [81, 168]. For instance, it is common to have tracking tasks in the objective function to specify that the end-effectors should follow a desired trajectory [118]. Unfortunately, designing and tuning the desired trajectories to realize complex tasks is time-consuming and often requires the expert knowledge of the planner and of the humanoid's kinematics and dynamics, which prevents an easy deployment of new tasks [123]. An alternative is to follow an imitation approach: a person performs a movement and the robot attempts to reproduce it [22]. Kinesthetic teaching is now a mature approach for robotic arms and industrial manipulators equipped with torque sensing: it allows the human operator to show the robot the desired trajectories by physically manipulating the robot links. While this approach is relatively straightforward for robotic arms, it can be hardly done for humanoids to demonstrate whole-body movements because it is not possible to physically guide all the robot links at once while ensuring the robot's balance. For this reason, teleoperation can be considered as an extension of the kinesthetic teaching concept at a whole-body level.

Teleoperation can be helpful for providing an intuitive way to realize complex tasks (eliminating the manual design or expert tuning problem), or even for conveying collaborative policies to humanoids. In this context, this thesis was funded under the European project AnDy [1] [82], that foresees robots being supplied with increasingly more ability to control physical collaboration with human partners through intentional interaction. For this matter, it becomes key to develop advanced human-like robot behaviors that can be easily interpreted by the human partners. Indeed, a consistent physical and social human-robot interaction is achieved when the robot is perceived by the human partners as "believable" and "purposeful" [139] through the way it moves, its appearance and through the consistency of its actions and behaviors [191]. Any physical or behavioral inconsistency with the human standards can be quickly perceived as "strange", making the robot become unacceptable for the human partners [116]. The human-likeness of the robot movements is, therefore, of crucial importance within the AnDy project and equally for this thesis.

1 Robot Teleoperation. An overview

The term **teleoperation** in general, indicates the **operation of a system (the robot) at a distance**, and is similar in meaning to the term "remote control".

Teleoperation was first conceived in the Argonne National Laboratory soon after World War II to handle radioactive materials with robot manipulators [67]. After that, its use has spread to multiple domains, attracting the attention of the research community. In 1980, Susumu Tachi proposed the idea of telexistence [176], a human-empowerment concept enabling humans to virtually

[1] Advancing anticipatory behaviors in Dynamic human-robot collaboration.

Figure 3: Examples of humanoid robot teleoperation. From top left to bottom right corner: [149], [43], [184], [80], [87], [17], [97], [11].

exist in another location where they could act freely in view of a more ecological and time-efficient society with an overall improved work-life balance. This concept has been developed through several research and development projects leading to the establishment of a company (Telexistence Inc.).

Through the years, the interest in the potential applications to a multitude of industrial fields of telexistence, and more in general of teleoperation, has increased. In the frame of this industrialization process of robot teleoperation and inspired by a visit to the Tachi Laboratory, the XPRIZE Foundation has recently launched the ANA Avatar XPRIZE, a four-year global competition started in January 2020, focused on the development of an avatar system that "will deploy the senses, actions, and presence of a human to a remote location in real time, leading to a more connected world."

This is not the first competition focusing on robot teleoperation. In fact in 2012, the U.S.

Figure 4: Environments where humanoid robots are more suitable than other robotic platforms. (credits: Jerry Pratt [15])

Department of Defense launched the DARPA Robotics Challenge (DRC) with the intention of promoting innovation in human-supervised robotic technology for disaster-response operations. The DRC finals also represented a midterm evaluation for WALK-MAN [14], a European Union Horizon 2020 project with the aim of building a humanoid teleoperated platform for inspection in dangerous unstructured environments damaged by natural events. The NASA Johnson Space Center's Val-EOD project is also focused on the use of teleoperated humanoid robots. The goal of the project is to develop humanoid robots, primarily NASA JSC's Valkyrie, to function as Explosive Ordnance Disposal operators [87].

Robot teleoperation has also been employed in space applications. The Canadarm system – officially Shuttle Remote Manipulator System (SRMS) – is a series of robotic arms that were used on the Space Shuttle orbiters to deploy, manoeuvre, and capture payloads [3]. It was first tested in orbit in 1981, on Space Shuttle Columbia's STS-2 mission. Its first operational use was on STS-3 to deploy and manoeuvre the Plasma Diagnostics Package, and has since flown on more than 90 space missions. Also the use of other teleoperated platforms has been investigated in the past decade. In 2018, experiments have been conducted with rovers and mobile manipulators being teleoperated from aboard the International Space Station (ISS), in the context of the projects METERON and Kontur-2 [107]. In 2011, the humanoid Robonaut 2 was sent to the ISS to help astronauts with various tasks [52] but stopped working in 2014 due to a short-circuit induced by the addition of new legs, and was finally sent back to Earth after 3 years. Also the robot Skybot F-850 has been rocketed to the ISS in August 2019 in an unmanned spacecraft carrying supplies [11]. The robot turned out to have a design that does not work well, demonstrating that there is still work to do to get humanoids in space.

More research effort is required to successfully control these kind of robots in real-world scenarios but still, they are the ideal candidates to replace humans in hazardous, remote, or inaccessible environments. In fact, differently from more conventional robotic platforms, the humanoids' structure is a better fit for environments and tasks that are designed for and performed by humans (see Figure 4) and their operational versatility makes them suitable for work activities that require a variety of complex movements, such as inspection, maintenance, and interaction with humans. In certain contexts such as telexistence, where other human partners and/or bystanders are expected to interact with the teleoperated robot, the human-likeness factor is even more important since it increases the acceptability and the engagement with the robot [53].

Many efforts of the robotics community have been devoted to enhance humanoids teleoperation and still many challenges have to be overcome, such as providing a better sense of presence to the human operator, designing intuitive and natural interfaces for humans, compensating the communication delays, or improving the control of the robot to achieve more dynamical motions and better interactions with other humans.

2 Contributions and Thesis Organization

The main objective of this thesis is, to develop a system and tools for teleoperating a humanoid robot. Our approach is illustrated Figure 5:

1. Defining an architecture and selecting proper devices for the teleoperation system
2. Translating the human motion into an equivalent feasible robot motion
3. Learning a generic "optimal" whole-body controller to replicate the human motion while preserving the robot's balance
4. Including a different higher-level modality of teleoperation for dynamic motions that cannot or should not be transferred from the human
5. Compensating for time delays in a non-ideal communication network, where the human operator can easily get confused by the round-trip delays between the commands sent to the robot and the corresponding visual feedback, failing to adequately control the robot.

Given the broad range of topics, for clarity of presentation, each chapter presents a state-of-the-art review of its relevant topics.

Chapter 1 reviews the systems and devices that have been adopted in the literature to teleoperate humanoids. We present the general architecture of a teleoperation system, together with the different sensing devices that can be used to capture the human motions, commands and/or states. Also the possible feedback interfaces that allow the human to perceive the robot states are presented. We then give an overview of the devices we used to teleoperate the humanoid robot iCub. Finally we review the different types of controllers that have been proposed in the literature to teleoperate humanoid robots.

Chapter 2 proposes methods to transfer the whole-body motion from the operator to the robot. We design a retargeting framework that allows the robot to replicate the motion of the human operator, acquired by a wearable motion capture suit, while maintaining the whole-body balance. We introduce some dynamic filter in the retargeting to forbid dangerous motions that can make the robot fall.

To test retargeting on the robot, we associate it with a control scheme based on a QP optimization solver. We validated our approach through several experiments on the iCub robot and simulations with the HRP-4C robot.

Chapter 3 investigates how to automatically learn an optimal controller configuration (soft and strict task priorities and convergence gains), looking for solutions that track a variety of desired task trajectories efficiently while preserving the robot's balance. We use multi-objective optimization to compare and choose among Pareto-optimal solutions that represent a trade-off of performance and robustness and can be transferred onto the real robot.

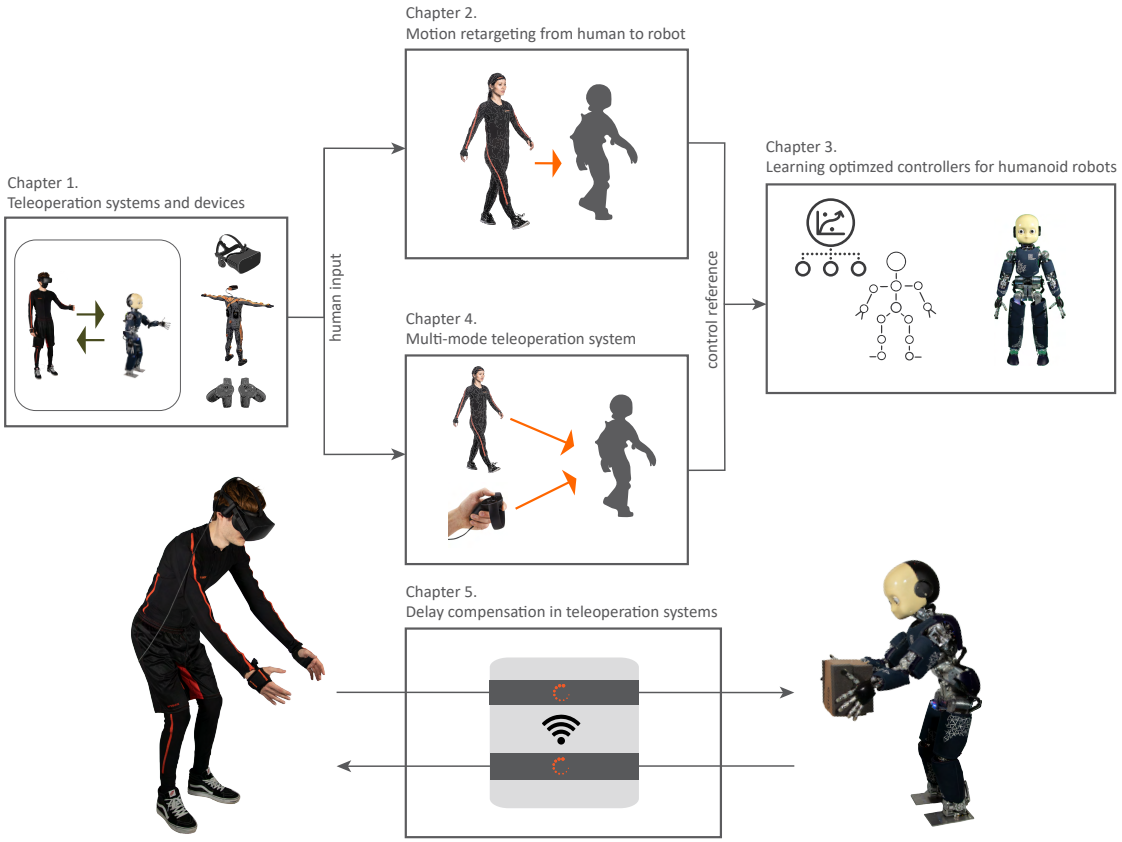


Figure 5: Thesis organization. In Chapter 1, we present the teleoperation systems and the teleoperation devices used nowadays and those we opted for. In Chapter 2 we show how the human information gathered from some of these devices can be translated into an equivalent reference motion for the robot. Then in Chapter 3 we present how to control the robot to optimally track the reference motion. In Chapter 4 we include a different mode of teleoperation for locomotion that uses a higher-level input rather than the human motion. Finally in Chapter 5 we present a solution for compensating the delays that are often encountered in non-ideal communication networks.

We experimentally validate our method by learning a control configuration for the humanoid iCub, to perform different whole-body tasks, such as reaching and picking up objects, squatting and opening doors.

Chapter 4 presents the possibility of using a multi-mode teleoperation system to handle differently those motions that can be transferred from human to robot and those which cannot or should not. The reason for the two distinct modes is that retargeting is essentially kinematic and it is not effective for on-line teleoperation of dynamic motions such as walking or stepping. In the presented system, the operator gives a direction and velocity references for the humanoid gait through the analog sticks. These are translated into a timed sequence of footsteps, swinging foot and center of mass trajectories through an MPC-based control scheme. When performing manipulations, the operator switches back to the full-control teleoperation mode.

The use case presented in the experiment consisted of guiding the walking of the robot with a joystick to reach a box on the ground, and then picking it up by fully controlling the whole-body of the robot.

Chapter 5 presents a delay compensation approach. A major problem preventing the deployment of teleoperation systems in real applications is the presence of communication delays between the human input and the feedback from the robot: even a few hundred milliseconds of delay can irremediably disturb the operator, let alone a few seconds. To overcome these delays, we introduce a system in which a humanoid robot *executes commands before it actually receives them*, so that the visual feedback appears to be synchronized to the operator, whereas the robot executed the commands in the past. To do so, the motions are parameterized using stacks of Probabilistic Movement Primitives, ProMPs [137]. the robot continuously predicts future commands based on the learned ProMPs that are trained on past trajectories and updated according to the last received commands.

In our experiments, an operator was able to successfully control the humanoid robot iCub with stochastic delays up to 2 seconds in several whole-body manipulation tasks, including reaching different targets, picking up, and placing a box at distinct locations.

2.1 Collaboration with the JRL

In this thesis, we also report the results obtained by collaborating with the CNRS-AIST Joint Robotics Laboratory (JRL) in Tsukuba, Japan, in the frame of the JSPS[2] Post-doctoral Fellowship Short-term Program. The goal of the program was to enhance the telexistence system used for the robot HRP-4C, in the context of the global competition ANA AVATAR XPRIZE. The JRL is participating at the competition as the team Janus and the candidate had work together with the team through the competition. However, due to the travel-restrictions imposed during the pandemic, the JSPS Fellowship was cancelled after an initial period of remote collaboration.

3 Publications

The body of work of this Ph.D. has produced several contributions in the form of academic articles, software, and videos, described below. For all the articles for which I am the first author, I was the main contributor to the project: I implemented the code, performed the experiments, analyzed the results with my supervisors, and wrote most parts of the article.

[2] Japan Society for the Promotion of Science

Accepted / Published Articles

– **Motion Retargeting from Human to Robot:** this work aimed to transfer motions from a human operator to a humanoid robot, making the robot replicate the human movements.

Contribution: we designed a retargeting framework that allows the robot to replicate the motion of the human operator, acquired by a wearable motion capture suit, while maintaining the whole-body balance. We introduced some dynamic filter in the retargeting to forbid dangerous motions that can make the robot fall. We validated our approach through several experiments on the iCub robot. The work is presented within Chapter 2.

> **L. Penco**, B. Clément, V. Modugno, E.M. Hoffman, G. Nava, D. Pucci, N.G. Tsagarakis, J.-B. Mouret, and S. Ivaldi. Robust real-time whole-body motion retargeting from human to humanoid. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 425–432. IEEE, 2018
>
> **Video** is available at: youtu.be/iZVAacyvYhM.

– **Multi-mode Teleoperation System:** this study proposes a teleoperation system that allows the operator to teleoperate both non-dynamic motions and dynamic motions that impact or leverage the robot dynamics and cannot be optimally retargeted from the operator.

Contribution: we present a system where the human operator can choose between two different modes using the buttons of the VR controller: a *low-level* mode for full real-time control of the robot via motion retargeting and a *high-level* mode to command walking or pre-optimized task trajectories. Both modalities share the same whole-body controller for computing in real-time the commands to be sent to the robot. The system is presented within Chapter 4.

> **L. Penco**, N. Scianca, V. Modugno, L. Lanari, G. Oriolo, and S. Ivaldi. A multimode teleoperation framework for humanoid loco-manipulation: An application for the iCub robot. *IEEE Robotics and Automation Magazine*, 26(4):73–82, 2019
>
> **Video** is available at: youtu.be/D3Uvle2WF5A.

– **Learning Optimized Controllers for Humanoid Robots:** this study proposes optimizing the parameters of a multi-task controller that can be used to control (teleoperate) a humanoid robot, obtaining a good tracking of the reference motion (human motion).

Contribution: we showed how to obtain robust solutions working on the real robot in a few trials that enable a good tracking of generic motions in double support. The study is presented within Chapter 3.

> **L. Penco**, E. M. Hoffman, V. Modugno, W. Gomes, J. Mouret, and S. Ivaldi. Learning robust task priorities and gains for control of redundant robots. *IEEE Robotics and Automation Letters*, 5(2):2626–2633, 2020
>
> **Video** is available at: youtu.be/RJW67SU6Yf0.

– **Review on Human-Humanoid interaction:** it reviews different aspects of human-humanoid interaction, such as social factors, robot interaction control, and human perception. Additionally it also reviews relevant applications on the field, such as humanoid used as avatars.

Personal Contribution: In this review, I was the main contributor for the sections concerning the introduction, the state-of-the-art on the control of the robot during interaction and the applications of humanoid collaborators.

L. Vianello, **L. Penco**, W. Gomes, Y. You, S.M. Anzalone, P. Maurice, V. Thomas, and S. Ivaldi. Human-humanoid interaction and cooperation: a review. *Current Robotics Reports*, 2:441–454, 2021

Under Revision

– **Survey on Teleoperation of Humanoid Robots:** it presents an overview of the teleoperation of humanoid robots with a system, control, human-centered perspective. We discuss different aspects of the topic, including technological and methodological advances as well as its applications.

Personal Contribution: In this review, I was the main contributor for the sections concerning the overview of humanoid teleoperation, control systems proposed for teleoperation and a co-contributor for sections regarding the technologies used for teleoperation and the potential applications of teleoperation. More in general, I compared the different options, advantages, and limitations and provided available examples in the literature in summary tables. Parts of the survey are also present in this introduction chapter.

K. Darvish, **L. Penco**, J. Ramos, R. Cisneros, J. Pratt, E. Yoshida, S. Ivaldi, and D. Pucci. Teleoperation of humanoid robots: A survey. 2022

– **Delay Compensation in a Teleoperation System with a Non-ideal Network:** it investigates a solution for compensating the delays in a non-ideal network while teleoperating the robot.

Contribution: In this work, we introduced a system in which the humanoid robot executes commands before it actually receives them, so that the visual feedback appears to be synchronized to the operator, whereas the robot executed the commands in the past. The study is presented within Chapter 5.

L. Penco, J.-B. Mouret, and S. Ivaldi. Prescient teleoperation of humanoid robots. *CoRR*, abs/2107.01281, 2021

Video is available at: youtu.be/N3u4ot3aIyQ.
The raw **dataset** is available at the DOI: 10.5281/zenodo.5913573.
C++ **code** of the Probabilistic Movement Primitive library is available at: github.com/hucebot/promp

1

Teleoperation Systems and Devices

In the literature, the terms teleoperation and telexistence have been used in different contexts. Telexistence, or telepresence, refers to the technology that allows the human to exist in a remote location through an avatar, experiencing real-time sensations from the remote site [176]. Similarly, the concept of teleoperation refers to a human operator remotely controlling a robot but the focus is mainly put on performing a dexterous task in the remote location. These terms are often used interchangeably. From another perspective, the teleoperation setup represents an interactive system where the robot imitates the human's actions to reach a common objective [69].

In a teleoperation setup, the user is the person who teleoperates the humanoid robot and identifies the teleoperation goal, i.e. intended outcome, through input devices. The devices represent the means of the interaction, the interface, between the user and the robot, which has to accomplish a variety of tasks. A task is the set of activities undertaken by the user and the robot in order to reach the specific goal [5]. The nature of the exchanging information, constraints, the task requirements, and the degree of shared autonomy determine different preferences on the interface modalities. Moreover, the choice of the interfaces should make the user feel comfortable, hence enabling a natural and intuitive teleoperation.

1.1 Teleoperation Architecture

The teleoperation system consists of several parts, including the perception of human states, their mapping to the robot, the robot controller, and the perception of the robot states and the remote environment, essential to give feedback to the user. Figure 1.1 is a schematic view of the architecture for teleoperating the robot. First, the human kinematic and dynamic information are measured in order to teleoperate the robot. In this regard, the high degrees of freedom of the robot and similar anthropomorphic geometry of human and humanoid robot suggest capturing the human motion and translating it into an equivalent one for the robot. There are cases in which also physiological signals are measured in order to estimate the psyhco-physiological state of the user, which can help enhancing the performance of the teleoperation. In some systems, the estimation of the user reaction forces/torques is required as well. The teleoperation system is employed not only for telemanipulation scenarios but also for social teleinteractions (i.e. remotely interacting with other people). In this case, the anthropomorphic motion of the robot and the robot social cues such as facial expressions can enhance the interaction experience. Therefore, rich human sensory information is indispensable.

On the basis of the measured states, the human-to-robot transferring policy – referred to as *retargeting* – is selected and the references are provided to the robot accordingly. Thus, the human

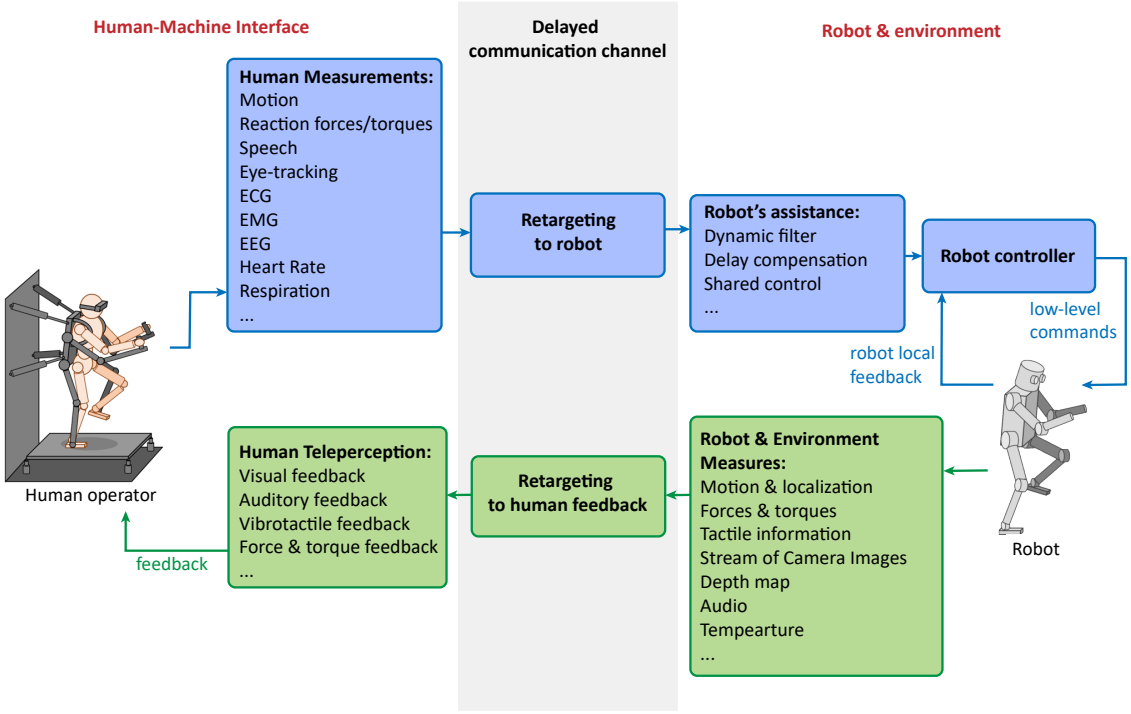


Figure 1.1: Schematic architecture of a complete humanoid robot teleoperation system.

sensory information is translated into a reference behaviour for the robot teleoperation. The motion retargeting methodology is presented in details in Chapter 2.

The retargeted information can be corrected by recurring to some dynamical filtering to ensure the balance of the robot as shown in Chapter 2, but also by using some robot autonomy in order to enhance the teleoperation experience of the operator or to improve the accuracy in the task execution. The latter approach is also referred to as *shared control* and is discussed later in the chapter within a dedicated section (Section 1.6.1).

To effectively teleoperate the robot, the user should make proper decisions; therefore he/she should receive different feedback from the remote environment and the robot. Also in this case the robot measurements have to be translated into meaningful feedback for the human, i.e. have to be retargeted on the human. For example, the forces measured on some parts of the robot's body may be used to make a vibro-tactile belt vibrate. In many cases, the sensors for perceiving the human data and the technology to provide feedback to the user are integrated together in a single interface.

Whether or not the feedback is coupled to the operator (or to the input device used to teleoperate the robot) is a significant feature that distinguishes one teleoperation technique from another. Bilateral teleoperation techniques directly couple operator and robot in position (velocity) and force through any of the so called bilateral control schemes: position-position (common error or symmetric position servo), force-position (force feedback), force-force (force servo-position or force-feedback servo). It is the objective of bilateral systems that the operator feels directly on its commanding hand (or any other part of the body used to control the robot) the contact force of the robot. Alternatively, in unilateral teleoperation systems, the coupling between operator and robot takes place only in one direction. The operator can still receive haptic feedback either as a kinesthetic cue not directly related to the contact force being generated by the robot, or as an

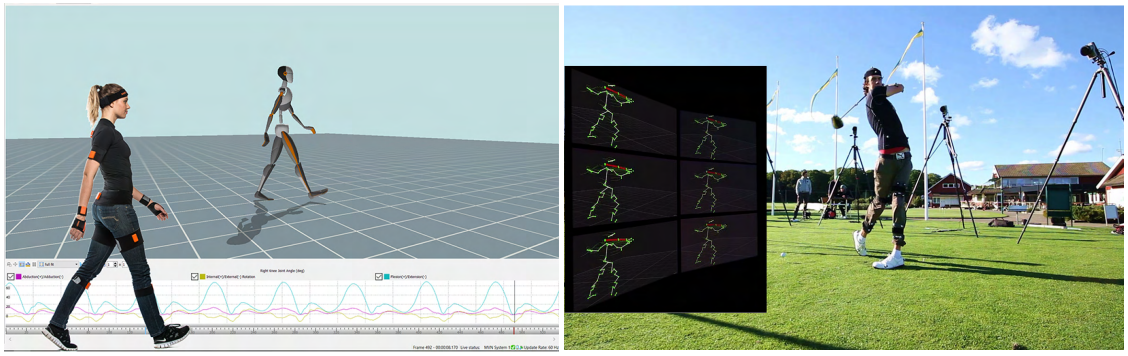


Figure 1.2: Examples of motion capture technologies. Left: wearable motion capture system based on the use of IMU sensors (credits: Xsens). Right: motion capture system based on optical tracking (credits: Reallusion).

indirect force in a passive part of her/his body not commanding the robot. The system presented in this thesis does not investigate the use of haptic feedback and uses a unilateral approach, which is the most proposed in humanoid robotics due to the difficulty of coupling the whole-body of the operator to that of the robot in bilateral systems. In the rest of the chapter, we will present different available interfaces and sensor technologies in teleoperation scenarios and explain the bilateral systems more exhaustively in a dedicated section.

The human retargeted information, together with the feedback from the robot, are streamed over a communication channel that could be non-ideal. In fact, long distances between the operator and the robot or poor network conditions may induce delays in the flow of information, packet loss and limited bandwidth, adversely affecting the teleoperation experience. We present the approaches proposed in the literature to teleoperate robots in such conditions in Chapter 5.

Finally, the robot local control loop generates the low level commands- i.e., joints position, velocity, or torque references- to the robot, taking into account the human references. Details about the different techniques proposed for controlling a teleoperated humanoid robot are given in Chapter 3.

1.2 Human Measurements Devices

In this section, we provide a detailed description of the technologies available to sense various human states, including the advances to measure the human motion, the physiological states, and the interaction forces with the environment. We report in TABLE 1.1 the various measurement devices adopted so far for humanoid robot teleoperation.

1.2.1 Human kinematics and dynamics measurements

To provide the references for the robot motion, we need to sense the human intentions. For simple teleoperation cases, the measurements may be granted through simple interfaces such as keyboard, mouse, or joystick commands [55]. However, for a more complex system like a humanoid robot, those simple interfaces may not be enough, especially when the user wants to exert a high level of control authority over the robot. Therefore, the need for natural interfaces for effectively commanding the robot arises. A solution is benefiting from the similarity of the human and the humanoid robot anthropomorphic geometries, i.e., providing the references to the robot limbs with

spatial analogies to those of the human (natural mapping). Therefore, the need to measure the human kinematics emerges.

Different technologies have been employed in the literature to measure the motion of the main human limbs, such as legs, torso, arms, head, and hands (Figure 1.2). An ubiquitous option are the Inertial Measurement Unit (IMU)-based wearable technologies. In this context, either a segregated set of IMU sensors are used throughout the body to measure the human motion or an integrated network of sensors is used [43, 184]. The former case normally provides the raw IMU values, while the latter provides the information of the human limb movements. In the second case, a calibration process computes the sensors transformations with respect to body segments [156]. This technology is especially of interest because of the high accuracy and frequency of the retrieved human motion information without the occlusion problem. However, its accuracy may suffer from the disturbances caused by the magnetic field and displacement of the sensors with respect to their initial emplacement.

Yet, another recent wearable technology to capture the hand pose is a stretchable soft glove embedded with distributed capacitive sensors [66]. A review about the textile-based wearable technology used to estimate the human kinematics is provided in [193].

Optical sensors are another technology used to capture the human motion, with active and passive variants. In the active case, the reflection of the pattern is sensed by the optical sensors. Some of the technologies include depth sensors and optical motion capture systems. Concerning the depth sensors, infrared patterns or laser beams are projected in the environment and the reflections are measured by an optical sensor. In the case of passive sensors, RGB monocular cameras (regular cameras) or binocular cameras (stereo cameras) are used to track the human motions. Thanks to the optical sensors, a skeleton of the human body is generated and tracked in 2-D or 3-D Cartesian space. The main problems with these methods are the occlusions and the low portability of the setup.

The previously introduced technologies can be used to measure the human gait information with a locomotion analysis. To obtain the gait information, normal or omnidirectional treadmills are employed in the literature [55]. While the treadmill can be used for even terrains, it would not work to retarget locomotions on uneven terrains. To respond this, a cockpit-like teleoperation setup has been recently proposed [80]. To estimate the interaction forces between the human and the teleoperation setup, force-torque sensors measuring the human wrenches can be integrated in ground plates, shoes, or exoskeletons. Richer information can be obtained by distributed capacitance sensors that measure the pressure manifold [172].

1.2.2 Human physiological measurements

Among the different sensors available to measure human physiological activities, we briefly describe those that have been mostly used in the teleoperation and robotics literature. An electromyography (EMG) sensor provides a measure of the muscle activity, i.e., contraction, in response to the neural stimulation action potential [173]. It works by measuring the difference between the electrical potential generated in the muscle fibres by employing two or more electrodes. There are two types of EMG sensors, the surface electromyography and the intramuscular electromyography. The former records the muscle activity from above the skin (therefore, noninvasive), while the latter measures the muscle activity by inserting needle electrodes into the muscle (intrusive). The main problem with EMG sensors, especially the surface one, is the low signal to noise ratio, which is the main barrier for a desirable performance [37]. The EMG signals have been used in the literature for teleoperating a robot or a prosthesis, for the estimation of the human effort and muscle forces [173], or for the estimation of the muscle stiffness. The EMG signals can give

Table 1.1: Teleoperation devices in the main works using humanoid robots.

ref	robot	teleoperation devices										
		human motion measurement				feedback						
						visual		haptic				
		motion-capture suit	optical-tracking	exoskeleton	mouse, keyboard, joystick, treadmill	display (GUI)	VR headset	WB exoskeleton	dual-arm exoskeleton	glove	balance	vibrotactile
[177]	TELESAR VI		✓				✓			✓		
[77]	TORO	✓										
[17]	TORO			✓					✓			
[79]	JAXON		✓				✓					
[80]	JAXON			✓			✓	✓				
[87]	Valkyrie				✓	✓						
[85]	Atlas				✓	✓						
[199]	DRC-HUBO				✓	✓						
[152]	HRP-2		✓									
[38]	HRP-2KAI				✓	✓						
[61]	HRP-4	✓										
[149]	little HERMES			✓			✓	✓			✓	
[143]	Fujitsu HOAP-3		✓								✓	
[32]	COMAN		✓								✓	✓
[97]	MAHRU	✓					✓					
[184]	iCub	✓			✓		✓					
[55]	iCub		✓		✓		✓					
[43]	iCub	✓			✓		✓					

Figure 1.3: Examples of interfaces used to get visual feedback from the robot. From left to right: Virtual Reality headset (CC BY-SA 3.0 IGO), Augmented Reality headset (credits: ŠKODA AUTO a.s. 2022) and Graphical User Interface [93].

information about the human motions with some anticipation by measuring the muscle activities within a few milliseconds in advance of the force generation; this could be exploited to anticipate the human operator's motion, enhancing the teleoperation.

Electroencephalography (EEG) sensors can be employed to identify the user mental state. They are most widely used in non-invasive brain-machine interface (BMI) and they monitor the brainwaves resulting from the neural activity of the brain. The measurement is done by placing several electrodes on the scalp and measuring the small electrical signal fluctuations [127]. Other sensors that could be employed in a telexistence scenario for an advanced estimation of the human phsychophysiological state include the Heart Rate Monitor sensors, which estimate the maximal uptake of the oxygen and the heart rate variability [19], useful to measure the user's fatigue, Electro-Oculography (EOG) sensors or video-based eye-trackers, which estimate of gaze position based on the pupil or iris position [174], important for the visual feedback given by Virtual Reality (VR) or Augmented Reality (AR) goggles, and capacitive thin-film humidity sensors, which measure the humidity of the gas-flow of the human skin [129], a good indicator of the human emotional stimuli and stress level.

1.3 Feedback Devices

A crucial point in robot teleoperation is to sufficiently inform the human operator of the states of the robot and its work site, so that she/he can comprehend the situation or feel physically present at the site, producing effective robot behaviors. TABLE 1.1 summarizes the different feedback devices that have been adopted for humanoid teleperation.

1.3.1 Visual feedback

A conventional way to provide situation awareness to the human operator is through visual feedback (Figure 1.3). Visual information allows the user to localize themselves and other humans or objects in the remote environment. Graphical User Interfaces (GUI) have been widely used by the teams participating at the DRC to remotely pilot their robots through the competition tasks [38, 85, 199]. Not only the information coming from the RGB cameras of the robot was displayed to the user but also other information such as depth, LIDAR, RADAR, and thermography maps of the remote environment. In the DRC, the operators were able to supervise the task execution through a task panel, using manual interfaces in case they needed to make cor-

Figure 1.4: Examples of interfaces used to get haptic feedback from the robot. From left to right: exoskeleton (credits: DFKI), vibro-tactile belt (credits: Biodex) and haptic gloves (credits: Facebook Reality Labs)

rections. The main window consisted of a 3D visualization environment of the robot's current state, together with perception sensor data, motion plans, and hardware driver status. Additionally, a teleoperation interface to support manual control by the operator was provided. A common approach adopted by the different teams was to guide the robot perception algorithms for point cloud fitting, used to estimate the 3D pose of objects of interest in the environment. In this case, the operator was annotating search regions by clicking on displayed camera images or by clicking on 3D positions in the point cloud.

An alternative way to give visual feedback to the human operator is to use VR headsets, connected to the robot cameras. Although this has demonstrated to be effective in several robotic experiments [43, 79, 97, 184], during locomotion tasks the user can get motion sickness, due to the fact that the images from the robot cameras are not stabilized, while the images perceived by the human eyes are automatically stabilized on the retina thanks to compensatory eye reflexes. This aspect could be improved by adopting digital image stabilization techniques or AR [160]. Another issue concerning the visual feedback is related to the limited bandwidth of the communication link, that can delay the stream of information. Considering that the human reaction time to visual information is approximately 250-300 ms, delays of this magnitude can intensify the user motion sickness and cause greater cognitive load [92].

1.3.2 Haptic feedback

Visual feedback is not often sufficient for many real-world applications, especially those involving power manipulation (with high forces) or interaction with other human subjects, where the dynamics of the robot, the contact forces with the environment, and the human-robot interaction forces are of crucial importance. In such scenarios also the haptic feedback is required to exploit the human operator's motor skills in order to augment the robot performance.

There are different technologies available in the literature to provide haptic feedback to the human (Figure 1.4). Force feedback, tactile and vibro-tactile feedback are the most used in teleoperation scenarios. The interface providing kinesthetic force feedback can be similar to an exoskeleton [101, 192] or can be cable driven. The latter only provides a tension force feedback, while the former provides the force feedback on different directions. Dual-arm exoskeletons have been proposed to guide the teleoperated robot during manipulation tasks while receiving haptic feedback through the same actuated exoskeleton arms [17, 176] and recently also a whole-body

exoskeleton cockpit has been proposed to teleoperate the JAXON robot during heavy manipulation tasks and stepping on uneven terrains [80], getting a force feedback on the whole limbs.

To convey the sense of touch, tactile displays have been adopted in the literature [190]. These can also provide temperature feedback to the user. Another widely used haptic feedback is the vibrotactile one, used as a sensory substitution to transmit senses of touch, forces, suggesting directions or to catch the attention of the user [32].

All these type of haptic feedback are combined in the telexistence system TELESAR V [176], which has been developed to provide complete cutaneous sensations to the human operator. The idea is that different physical stimuli give rise to the same sensation in humans and are perceived as identical. This is due to the the fact that human skin has limited receptors and can perceive only force, vibration, and temperature, which in [175] are defined as *haptic primary colors*. The definition derives from the fact that humans perceive the light of different spectra as having the same color if the light has the same proportion of the different spectral components. This is because the human retina typically contains three types of color receptors called cone cells or cones, each of which responds to a different range of the color spectrum. Humans respond to light stimuli via 3D sensations, which generally can be modeled as a mixture of the three primary colors (red, green, and blue). Much like the human retina, human skin has limited receptors. In physiological space, cutaneous perception is created through a combination of nerve signals from several types of tactile receptors located below the surface of the skin. If each activated haptic receptor is considered as a sensory base, any given pattern of cutaneous sensation could be replicated through synthesis by using these bases. Objects with the same force, vibration and temperature are perceived as the same, even if their physical properties are different. It is thus sufficient to combine these *colors* in order to reproduce any cutaneous sensation without actually touching the real object.

1.3.3 Balance feedback

Haptic feedback can also be used to transmit to the operator a sense of the robot's balance. The idea behind the balance feedback is to transfer to the operator the information about the *effect of disturbances* over the robot dynamics or stability instead of directly mapping to the human the disturbance forces applied to the robot. In [32], Brygo *et al.* proposed to provide the feedback of the robot's balance state by means of a vibro-tactile belt. Based on the distance of the CoP from the support polygon boundaries, a directional signal warns the operator when the stability margin of the robot drops. The operator is expected to modify his stance so as to reestablish the balance of the robot.

Peternel and Babic [143] propose a cable-driven haptic interface that maps the state of the robot's balance to the human demonstrator by exerting forces that are equivalent to the position of the robot's CoP. Alternatively, Abi-Farrajil *et al.* [17] introduced a task-relevant haptic feedback interface composed by two light-weight robotic arms that receives high-level informative haptic cues informing the user about the impact of her/his potential actions on the robot's balance. These studies do not investigate the case of dynamic behaviors, but are rather limited to double support scenarios. In [149], simultaneous stepping is enabled via bilateral coupling between the human operator, wearing a Balance Feedback Interface (BFI), and the robot. The BFI is composed of an passive exoskeleton which captures human body posture and a parallel mechanism which applies feedback forces to the operator's torso. The approach focuses on synchronizing the Divergent Component of Motion (DCM) trajectory of the human and that of the robot by imposing dynamic similarity between the two using the force feedback.

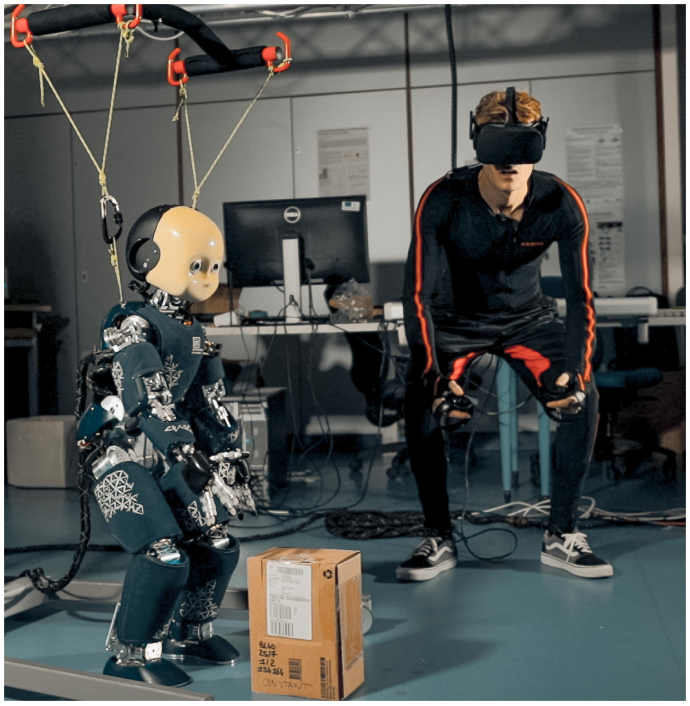


Figure 1.5: Employed teleoperation materials. The user teleoperating the iCub robot is equipped with the Xsens suit MVN Link and the Oculus Rift VR headset.

1.3.4 Auditory feedback

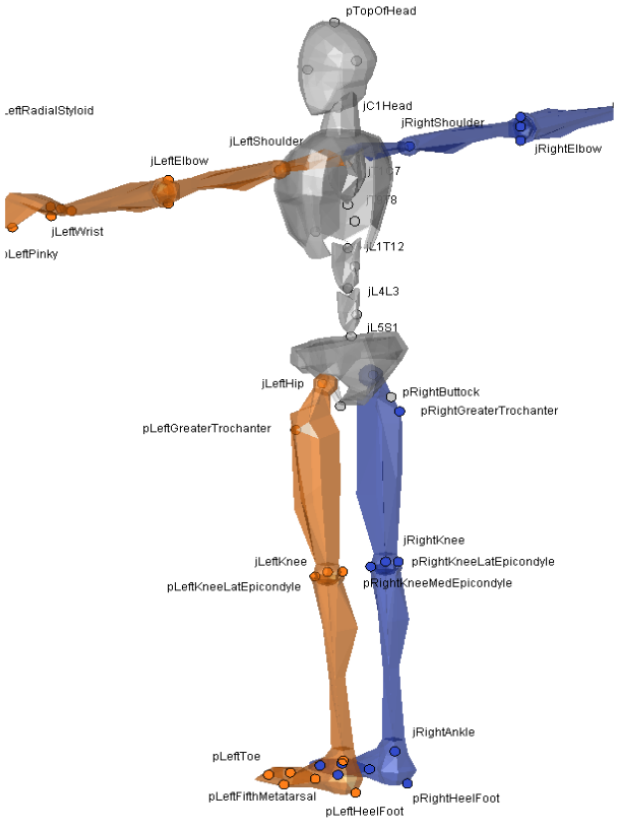
Auditory feedback is another means of communication. It is mainly provided to the user through headphones, single or multiple speakers. Auditory information can be used for different purposes: to enable the user to communicate with others in the remote environment through the robot, to increase the user situational awareness, to localize the sound source by using several microphones or to detect the collision of the robot links with the environment [12]. The user and the teleoperated robot may also communicate through the audio channel, for example for state transitions.

1.4 Employed Teleoperation Material

In this section we present the devices that we used in our teleoperation system and the robots that we teleoperated in the experiments.

1.4.1 Xsens motion capture system

Recent developments in human motion capture allow now high-fidelity and high-frequency tracking data. Motion-capture is widely used nowadays in various fields including physiotherapy, surveillance, computer graphics and foremost in the movie industry, using external cameras or wearable sensors. In our experiments, we used the Xsens MVN system [156]. Xsens MVN solutions are based on inertial technology, using gyroscopes, accelerometers, and magnetometers rather than digital cameras, volumes and markers to capture movement. The hardware version of the Xsens MVN product line we used in all our experiments was the MVN Link (Figure 1.5). It consists of a lycra suit with 17 wired trackers, a wireless data link and a 9.5h life battery.



Number	Segment Label	Tracker	Joint
1	Pelvis	Pelvis	jL5S1
2	L5	T8	jL4L3
3	L3	Head	jL1T12
4	T12	RightShoulder	jT9T8
5	T8	RightUpperArm	jT1C7
6	Neck	RightForeArm	jC1Head
7	Head	RightHand	jRightC7Shoulder
8	Right Shoulder	LeftShoulder	jRightShoulder
9	Right Upper Arm	LeftUpperArm	jRightElbow
10	Right Forearm	LeftForeArm	jRightWrist
11	Right Hand	LeftHand	jLeftC7Shoulder
12	Left Shoulder	RightUpperLeg	jLeftShoulder
13	Left Upper Arm	RightLowerLeg	jLeftElbow
14	Left Forearm	RightFoot	jLeftWrist
15	Left Hand	LeftUpperLeg	jRightHip
16	Right Upper Leg	LeftLowerLeg	jRightKnee
17	Right Lower Leg	LeftFoot	jRightAnkle
18	Right Foot		jRightBallFoot
19	Right Toe		jLeftHip
20	Left Upper Leg		jLeftKnee
21	Left Lower Leg		jLeftAnkle
22	Left Foot		jLeftBallFoot
23	Left Toe		

Figure 1.6: Xsens skeleton with associated anatomical landmarks and joints. Left: Selection of anatomical landmarks as is used in MVN. Right: number of each segment, tracker and joint, as is used in MVN. Images taken from the user manual [157].

Figure 1.7: The iCub robot.

The software application MVN Animate can be used in conjunction with Xsens MVN hardware to record, monitor and analyze the movements. The system captures the movement and digitizes it, generating as output a 3D model of the human skeleton (Figure 1.6) that reproduces the movements performed by the person. The MVN Fusion Engine calculates the position and orientation, and other kinematic data of each body segment with respect to an earth-fixed reference co-ordinate system. By default, the earth-fixed reference coordinate system used is defined as a right handed Cartesian coordinate system. The axes of each body frame are aligned with this global reference frame. The recording session saves the tracked data containing the information of the tracked body segments (their pose, velocity and acceleration) together with the joint angles values and the center of mass position; all the information that is generally used in a retargeting problem. Figure 1.6 contains a list of the tracked body segments and joints. Further details about the system can be found in [156].

1.4.2 Oculus virtual reality system

In our experiments we employed the Oculus Rift headset. It uses a pair of screens that display two images side by side, one for each eye. A set of lenses is placed on top of the panels, focusing and reshaping the picture for each eye, and creating a stereoscopic 3D image. The goggles have embedded sensors that monitor the wearer's head motions and adjust the image accordingly. The latest version of the Oculus Rift is bolstered by an external positional-tracking sensor, which helps track head movements more accurately. The end result is the sensation that you are looking around a 3D world. Rift specs include a 2160 x 1200 OLED display which delivers 1080p per eye, a 110-degree field of view with a 90Hz refresh rate. The device also features a built in accelerometer, gyroscope, magnetometer and 360-degree positional tracking that follows six axes of movement. The Rift also has associated a pair of joystick that can be used to interact with the virtual reality, both by pressing buttons and moving analog sticks.

1.4.3 The iCub humanoid robot

The robotic platform we used in our teleoperation system is the iCub robot (Figure 1.7). iCub [126] is a research-grade open-source humanoid robot designed by the Italian Institute of Technology (IIT) to experiment with embodied artificial intelligence. It measures 104 cm in height and weighs 22 kg, which roughly correspond to the body dimensions of a five year old child. iCub has 53 actuated degrees of freedom: 7 in each arm, 9 in each hand (3 for the thumb, 2 for the index, 2 for the middle finger, 1 for the coupled ring and little finger, 1 for the adduction/abduction), 6 in the head (3 for the neck and 3 for the cameras), 3 in the torso/waist, 6 in each leg. In this thesis, we did not use hands and we did not move the eyes, we therefore controlled 32 degrees of freedom. The head has stereo cameras in a swivel mounting in the corresponding location of the human eye sockets. iCub is also equipped with six 6-axial force/torque (F/T) sensors in the upper arms, legs and feet, an IMU in the head, and a distributed tactile skin.

1.4.4 The HRP-4C humanoid robot

The HRP-4C [91], nicknamed Miim, is a feminine-looking humanoid robot created by the National Institute of Advanced Industrial Science and Technology (AIST). It measures 158 centimetres in height and weighs 43 kilos including a battery pack. It has a realistic head and face made of artificial skin. The robot has 42 actuated degrees of freedom: 6 in each arm, 2 in each hand (1 for the thumb, 1 for the fingers), 3 in the neck, 3 in the waist, 6 in each leg, and 8 in the face that allow it to make some facial expressions. In our simulations, we did not use the hands and the facial joints, we therefore controlled 30 degrees of freedom. During our collaboration with AIST, we could not work with the real robot located in Tsukuba (Japan) due to pandemic-related travel restrictions. Hence, only results obtained in simulation are reported in this thesis.

1.5 Teleoperation Controllers for Humanoid Robots

One of the main challenges related to the whole-body control of teleoperated humanoid robots, is to guarantee the dynamical balance of the robot while trying to follow the retargeted human references. This requires that the dynamics of the robot (or of a simplified model of the robot) has to be included at least in either the control law or in the computation of the reference trajectories, i.e., in the motion retargeting problem (Chapter 2). Different control strategies have been proposed in the literature.

1.5.1 Quadratic Programming control

One common approach is to formalize the controller as a multi-task constrained Quadratic Programming (QP) problem with inequality and equality constraints, where the references are the retargeted human inputs. The choice of tasks is generally standard. The position of the center of mass is taken into consideration to encode the balance of the robot. Cartesian tasks to control the pose of a specific link are widely used. The end-effectors poses are essential to perform any manipulation or locomotion tasks. To provide more stability, a task concerning the orientation of the floating base is commonly included in the controller. Alternatively, the orientation of other links, such as the head or the chest, can be considered. A postural task is also often considered to make the robot move in a more human-like and legible way.

For each task $\mathcal{T}_k$, a reference $\boldsymbol{b}_k$ (i.e., the retargeted human input) is defined, which is a function $f(\lambda_k \boldsymbol{e}_k)$ of the task error $\boldsymbol{e}_k$ and a parameter that has to be tuned, which we denote as feedback

gain (FG) λ_k. When multiple tasks are considered, they are organized according to different levels of priority, which can be strict priorities [49, 96, 134], i.e. *null-space relations*, and/or the soft task priorities [115, 161], i.e. *weighted combinations*. The set of n tasks $\{\mathcal{T}_1, \ldots, \mathcal{T}_n\}$ organized according to different soft or strict priorities, is usually referred to as stack $\mathcal{S}$:

$$\begin{aligned} \mathcal{S} = (w_1\mathcal{T}_1 + ... + w_i\mathcal{T}_i)/ \\ \vdots \\ (w_j\mathcal{T}_j + ... + w_n\mathcal{T}_n); \end{aligned} \tag{1.1}$$

where $\mathcal{T}_a + \mathcal{T}_b$ means that the tasks $\mathcal{T}_a$ and $\mathcal{T}_b$ are in a soft priority relation with w_a, w_b being the corresponding weights, while $\mathcal{T}_a/\mathcal{T}_b$ means that the two tasks are in a strict priority relation ($\mathcal{T}_b$ acts in the null space projection of $\mathcal{T}_a$).

For a given task $\mathcal{T}_k$, the general QP formulation consists in solving the following optimization problem at each control time step:

$$\begin{aligned} \min_{\boldsymbol{u}} \; & \|\boldsymbol{A}_k\boldsymbol{u} - \boldsymbol{b}_k\|^2 + \epsilon\|\boldsymbol{u}\|^2 \\ s.t. \quad & \underline{\boldsymbol{c}} \leq \boldsymbol{C}\boldsymbol{u} \leq \overline{\boldsymbol{c}} \\ & \boldsymbol{u}_{min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{max} \end{aligned} \tag{1.2}$$

where $\boldsymbol{u}$ is the control input vector; $\boldsymbol{A}_k$ is the equivalent Jacobian matrix of the task k; $\boldsymbol{b}_k$ the reference value for the task, ϵ a regularization factor used to handle singularities, $\boldsymbol{u}_{min}$ and $\boldsymbol{u}_{max}$ the control input limits and $\underline{\boldsymbol{c}} \leq \boldsymbol{C}\boldsymbol{u} \leq \overline{\boldsymbol{c}}$ are other equality and inequality constraints, e.g. dynamics, collision avoidance, and contact related constraints.

If we consider n levels of hierarchies (strict priorities), n QP problems can be solved including local equality constraints to ensure the strict priorities between the tasks in each hierarchy i, i.e. $\boldsymbol{A}_{i-1}\boldsymbol{u}_{i-1} = \boldsymbol{A}_{i-1}\boldsymbol{u}$, ... , $\boldsymbol{A}_0\boldsymbol{u}_0 = \boldsymbol{A}_0\boldsymbol{u}$. There are also other ways that solve hierarchical problems using a single QP instead of a cascade of QPs [58]. At a given level i of the hierarchy, we can also consider a weighted combination of tasks:

$$\begin{aligned} \min_{\boldsymbol{u}} \; & \sum_k w_k(\|\boldsymbol{A}_k\boldsymbol{u} - \boldsymbol{b}_k\|^2) + \epsilon\|\boldsymbol{u}\|^2 \\ s.t. \quad & \underline{\boldsymbol{c}} \leq \boldsymbol{C}\boldsymbol{u} \leq \overline{\boldsymbol{c}} \\ & \boldsymbol{u}_{min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{max} \end{aligned} \tag{1.3}$$

The QP formalism has become widespread not only because it allows one to specify multiple tasks but also because it can be applied to different types of control [17, 43, 61, 77, 87, 184].

Inverse Dynamics control

When the control input $\boldsymbol{u}$ is the robot torques or joint accelerations, the QP is computing the Inverse Dynamics (ID) of the robot. A common situation is to minimize the torques and joint accelerations of the robot in order to follow retargeted end-effector poses, without violating the robot constraints:

$$\underset{\dot{\boldsymbol{v}},\boldsymbol{\tau}}{\text{minimize}} \; \|\ddot{\boldsymbol{p}} - \ddot{\boldsymbol{p}}^d\|^2 \tag{1.4}$$

$$s.t. \quad \boldsymbol{M}\dot{\boldsymbol{v}} + \boldsymbol{C}\boldsymbol{v} + \boldsymbol{g} = \boldsymbol{B}\boldsymbol{\tau} + \sum \boldsymbol{J}^\top \boldsymbol{f}_{ext} \tag{1.5}$$

$$\boldsymbol{\tau}_{min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{max} \tag{1.6}$$

$$\tfrac{\boldsymbol{v}_{min} - \boldsymbol{v}}{\Delta t} \leq \dot{\boldsymbol{v}} \leq \tfrac{\boldsymbol{v}_{max} - \boldsymbol{v}}{\Delta t} \tag{1.7}$$

$$\tfrac{\boldsymbol{q}_{min} - \boldsymbol{q} - \boldsymbol{v}\Delta t}{\frac{1}{2}\Delta t^2} \leq \dot{\boldsymbol{v}} \leq \tfrac{\boldsymbol{q}_{max} - \boldsymbol{q} - \boldsymbol{v}\Delta t}{\frac{1}{2}\Delta t^2} \tag{1.8}$$

$$\boldsymbol{J}_C\dot{\boldsymbol{v}} + \dot{\boldsymbol{J}}_C\boldsymbol{v} = 0 \tag{1.9}$$

$$\boldsymbol{f}_C \geq 0 \tag{1.10}$$

$$\ddot{d}(b_1, b_2) \geq \frac{1}{\Delta t}(-\beta\frac{d(b_1,b_2)-\delta_s}{\delta_i-\delta_s} - \dot{d}(b_1, b_2)) \tag{1.11}$$

where $\boldsymbol{v} = \dot{\boldsymbol{q}}$ are the joint velocities and $\dot{\boldsymbol{v}} = \ddot{\boldsymbol{q}}$ the joint accelerations, $\ddot{\boldsymbol{p}}$ is the reference (teleoperated) value for the task expressed in accelerations, (1.5) is the dynamics equation of motion of the robot, (1.6) the torque limits, (1.7) the joint velocity limits, (1.8) the joint limits, (1.9) is the non-sliding contact constraint, (1.10) the friction cone constraint, (1.11) the collision-avoidance constraint between body b_1 and b_2 of the robot or the environment, with distance between the two bodies $d(b_1, b_2)$, damping coefficient β, security distance δ_s and influence distance δ_i. The inverse dynamics (ID) approach (1.4) has been proposed in [61] to handle multi-contact teleoperated motions on the HRP-4 robot. The reference values can alternatively be expressed in the task space, recurring to task-space inverse dynamics (TSID), which has been proposed for teleoperating the HRP-2 robot in [152], during a live performance consisting of fine balanced captured dance movements.

Momentum-based control

Also momentum-based controller can be designed in a QP fashion [100]. According to Euler's laws of motion, the rate of change of centroidal momentum is equal to the sum of all external wrenches applied to the robot:

$$\dot{\boldsymbol{h}} = \boldsymbol{f}_g + \sum_i \boldsymbol{f}_{gr,i} + \sum_i \boldsymbol{f}_{ext,i} \tag{1.12}$$

where $\boldsymbol{f}_g = (\boldsymbol{0}_{3\times1}^\top, mg^\top)$ is the wrench due to gravity, $\boldsymbol{f}_{gr,i}$ are ground reaction wrenches exerted upon the robot's links due to contact between these links and the environment, and the $\boldsymbol{f}_{ext,i}$ are other external wrenches applied to the robot. Since there is an affine relationship between the robot joint velocities and the robot's centroidal momentum

$$\boldsymbol{h} = \boldsymbol{A}(\boldsymbol{q})\boldsymbol{v}, \tag{1.13}$$

equation (1.12) can be written as

$$\boldsymbol{A}\dot{\boldsymbol{v}} + \dot{\boldsymbol{A}}\boldsymbol{v} = \boldsymbol{f}_g + \boldsymbol{Q}\boldsymbol{\rho} + \sum_i \boldsymbol{f}_{ext,i} \tag{1.14}$$

where the sum of all ground reaction wrenches has been written in a single matrix equation, with $\boldsymbol{Q}$ being the transformation matrix from a task frame to the centroidal frame and $\boldsymbol{\rho}$ the vector of ground reaction wrenches expressed in the task frame. From (1.14) we can formulate the following QP problem:

$$\begin{aligned} \underset{\dot{\boldsymbol{v}}_d,\boldsymbol{\rho}}{\text{minimize}}\ & (\boldsymbol{A}\dot{\boldsymbol{v}}_d - \boldsymbol{b})^\top \boldsymbol{C}_h(\boldsymbol{A}\dot{\boldsymbol{v}}_d - \boldsymbol{b}) + \boldsymbol{\rho}^\top \boldsymbol{C}_\rho \boldsymbol{\rho} + \dot{\boldsymbol{v}}_d^\top \boldsymbol{C}_q \dot{\boldsymbol{v}}_d \\ s.t.\quad & \boldsymbol{A}\dot{\boldsymbol{v}}_d + \dot{\boldsymbol{A}}\boldsymbol{v} = \boldsymbol{f}_g + \boldsymbol{Q}\boldsymbol{\rho} + \textstyle\sum_i \boldsymbol{f}_{ext,i} \quad (1.14) \\ & \boldsymbol{\rho}_{min} \leq \boldsymbol{\rho} \leq \boldsymbol{\rho}_{max} \end{aligned} \tag{1.15}$$

where $\boldsymbol{C}_h$, $\boldsymbol{C}_\rho$ and $\boldsymbol{C}_q$ are weighting matrices and $\boldsymbol{b} = \dot{\boldsymbol{h}}_d - \dot{\boldsymbol{A}}\boldsymbol{v}$, with $\dot{\boldsymbol{h}}_d \in \mathbb{R}^6$ being a desired rate of change of centroidal momentum. Finally the pre-specified external wrenches $\boldsymbol{f}_{ext,i}$, the computed $\boldsymbol{f}_{gr,i}$ ground reaction wrenches and joint accelerations $\ddot{\boldsymbol{q}}_d$ are used as the input to a

recursive Newton–Euler inverse dynamics algorithm to compute the robot joint torques from (1.5). This approach has been adopted to teleoperate through user interfaces both the Atlas and Valkyrie robots [86, 87]. In [43], Darvish *et al.* also propose the use of a momentum-based control but for the teleoperation of the iCub robot during single support motions. The control problem is formulated as QP optimization problem to achieve the two tasks of maintaining the balance and following the retargeted human posture, while carefully monitoring and regulating the contact wrenches, considering the associated feasible domains.

Inverse Kinematics control

An alternative to torque control is to control the robot in position casting an inverse kinematics (IK)-based QP to compute the joint angles commands:

$$\begin{aligned} \underset{\boldsymbol{v}}{\text{minimize}} \; & \sum_k w_k((\boldsymbol{J}_k\boldsymbol{v} - \boldsymbol{b}_k)^\top \boldsymbol{C}_q(\boldsymbol{J}_k\boldsymbol{v} - \boldsymbol{b}_k) + \boldsymbol{v}^\top \boldsymbol{C}_\epsilon \boldsymbol{v}) \\ s.t. \quad & \underline{\boldsymbol{d}} \leq \boldsymbol{D}\boldsymbol{v} \leq \overline{\boldsymbol{d}} \\ & \boldsymbol{v}_{min} \leq \boldsymbol{v} \leq \boldsymbol{v}_{max} \end{aligned} \tag{1.16}$$

where $\boldsymbol{C}_q$ and $\boldsymbol{C}_\epsilon$ are positive definite cost matrices and $\boldsymbol{b}_k = \dot{\boldsymbol{p}}_d + \lambda_k \boldsymbol{e}$ is the reference value for the k-th task, with $\dot{\boldsymbol{p}}_d$ being the feed-forward velocity term, λ_k the Feedback Gain and $\boldsymbol{e}$ the Cartesian pose error. Equation (1.16) can also be written as:

$$\begin{aligned} \arg\min_{\dot{\boldsymbol{q}}} & \textstyle\sum_i w_i f_i + \sum_j w_j g_j + \boldsymbol{C}\dot{\boldsymbol{q}} \\ & f_i = ||\boldsymbol{J}_i\dot{\boldsymbol{q}} - \dot{\boldsymbol{x}}_i||^2 \\ & g_j = ||\dot{\boldsymbol{q}}_j - \dot{\boldsymbol{q}}_j^r||^2 \\ & \text{subject to } \; \boldsymbol{J}\dot{\boldsymbol{q}} = \dot{\boldsymbol{x}} \\ & \boldsymbol{A}\dot{\boldsymbol{q}} \leq \boldsymbol{b} \end{aligned} \tag{1.17}$$

In this case, the cost function consists of terms f_i with relative weight w_i concerning the pose of a specific body link i, where $\boldsymbol{J}_i$ is the Jacobian matrix for body link i and $\dot{\boldsymbol{x}}_i = \dot{\hat{\boldsymbol{y}}}_i$ are the reference velocities for body link i. Additionally terms g_j with relative weight w_j concern the posture of certain joints j, where $\dot{\boldsymbol{q}}_j^r = \dot{\hat{\boldsymbol{y}}}_j$ are the reference joint velocities for joints j. $\boldsymbol{C}\dot{\boldsymbol{q}}$ is a regularization term used to get a unique solution and avoid singularities, where $\boldsymbol{C}$ is a linear cost vector. The equality constraints correspond to the highest priority task, which should be solved exactly.

This approach is the main one we used in our teleoperation system with the iCub robot. It has been widely used in the literature [55,77,79,97,199] to teleoperate robots in position, since it does not require motors equipped with torque sensors and is the easiest to implement and debug. We remind that iCub is position-controlled and it is not provided with joint torque sensors that enable high-frequency torque control. It is possible to estimate the joint torques [83] but the control performance using this estimation as local torque measure were not good enough for complex motions. Since the QP control law does not include any information about the dynamics of the robot, this has to be considered when computing the reference trajectories for the controller (as we have shown in Chapter 2). When trying to make the robot imitate the human stepping for example, the impact force at the ground on the humanoid robot's foot can be a serious external disturbance. In [79], Ishiguro *et al.* implement a safety mechanism for a soft and safe ground contact, by setting a vertical velocity limitation that is proportional to the foot height. After processing the human input references, they modify end-effector poses and center of mass position to preserve the robot balance based on the concept of Divergent Component of Motion (DCM) [56] and feed them into an IK solver. Other approaches use the LIPM and/or the DCM to compute dynamically meaningful control references for the locomotion of the robot [55, 77, 89] and feed them to a IK solver, as we will see in more details in the next Chapter 4.

Feedback controllers

In the literature, it is common to see position-controlled robots employing an additional feedback controller, also referred to as *stabilizer*, to correct the control references to work against many existing disturbances in the real system [38, 77, 79, 199]. For example in [77], the CoM-ZMP controller proposed by Choi *et al.* [36] is adopted as a feedback balance controller for an online human walking imitation. The CoM-ZMP stabilizer proposed by Choi *et al.* [36], has also been adopted to teleoperate the robot MAHRU in [97], which first proposed a simultaneous assignment of upper-body tasks and walking motion. The online footprint imitation is realized by feeding the recognized and adapted human step parameters into a classical ZMP based walking controller. In [79] the final retargeted joint angles are stabilized by using a bipedal feedback controller [89], which acts as an external disturbance absorber using the feet force sensors values and reference ZMP trajectories. With this additional control, the robot JAXON has been successfully teleoperated while performing very dynamic motions like high-kicking or hitting a tennis ball with a racket. In [152], the TSID control, framed as a hierarchical QP, handles the effects of the dynamics in feed-forward, with a feedback loop coming from the commercial stabilizer of HRP-2, which controls the movements of the flexible parts of the feet according to the measurements of the ankle force-torque sensors.

1.5.2 Other control formalisms

Solving a QP at each control cycle to compute the control inputs is not the only viable solution. An alternative can be to rely on a full-state linear quadratic regulator (LQR), using the full dynamics of the robot [120]. The dynamics of the robot (1.5) is numerically linearized, determining a linear time invariant system that leads to an optimization problem in the reduced state space. It is possible to optimize for elements in the task space defined as a linear combination of the original state vector, and choose the task space objectives to be the regulation of the center of mass and total momentum or other references coming from the operator controlling the robot [119].

Another control strategy that can be adopted to teleoperate humanoid robots is model predictive control (MPC). MPC is an optimization-based control technique applied to dynamical systems. MPC optimizes an objective function based on a predicted evolution of the system state [30]. The use of MPC for real time teleoperation is typically not straightforward and it is usually convenient to recur to a simplified prediction model. Generally, for robot walking, the LIPM or extended versions of it [145, 165] are adopted to implement a guided walking of the robot, as we describe in more details in Chapter 4.

Another possible control strategy is to adopt a posture and balance controller based on contact force optimization [133]. After defining the net forces and moments the robot must apply on its own CoM in order to follow the reference human motion, the net forces and the knowledge of the contact state are used to generate the proper force distribution to each leg (or other contact points, if available). The robot follows the human movements while maintaining balance by controlling the interaction forces between its feet and the ground. In their bilateral system, Ramos *et al.* [149] use a similar strategy, teleoperating the robot little HERMES while stepping.

1.6 Enhanced Teleoperation Strategies

In many teleoperation scenarios, the tasks can be performed more efficiently by sharing the autonomy among the human and robot or by directly coupling the human input with the robot state. This section provides more details about shared control and bilateral strategies for teleoperation.

1.6.1 Shared control

In this approach, the operator's input may be modified using some metric to enhance performance or safety by sharing the control authority between the robot and the operator [53]. In shared-control teleoperation, the robot assists the user in accomplishing the desired task, potentially making teleoperation easier and more seamless. For example in [148], Rakita *et al.* teleoperate the upper-body of a humanoid using a shared-control approach, providing on-the-fly assistance to help the user complete bimanual tasks more easily. They use an optimization-based IK solver, which is able to handle trade-offs between many objectives, such as smooth joint motion, kinematic singularity avoidance, or self-collision avoidance. In many shared-control approaches, the user provides an input $\boldsymbol{u}$, which enables the robot to take into account the human intent, and assist them in the task by adjusting the motion or by executing a pre-optimized version of that motion. Then, a policy blending arbitrates the user input $\boldsymbol{u}$ and the enhanced robot motion $\boldsymbol{r}$, determining the final reference:

$$\boldsymbol{u}^* = (1-\alpha)\boldsymbol{u} + \alpha\boldsymbol{r} \tag{1.18}$$

where α can be any function. A common choice is the confidence in the prediction of the user intent:

$$\alpha = max(0, 1 - d/D) \tag{1.19}$$

where d the distance to the goal and D some threshold past which the confidence is 0. In this case the closer the robot gets to a predicted goal, the more likely that goal is the correct one, and the input $\boldsymbol{r}$ is preferred over $\boldsymbol{u}$. Differently, in haptic-based shared-control approaches, only the user input actuates the robot but haptic information is used to guide the user during the task execution [16, 146, 167] or to enhance the user comfort during the telemanipulation [147].

1.6.2 Bilateral systems

Bilateral teleoperation techniques have been widely used in the literature for robot manipulators. In this approach, not only the robot receives kinodynamic references from the human, but the operator also receives kinesthetic feedback (force, pressure, vibration, etc) from the machine. This feedback informs the operator about the robot's performance reproducing the commanded motion or about external disturbances applied to the machine. The approaches described next focus on teleoperating solely the upper-body or on coupling the human operator and the machine at a whole-body level.

Upper-body bilateral teleoperation (admittance control)

A simple strategy that has been adopted on humanoid robots consists in teleoperating the upper-body in a bilateral fashion, while using a separate balancing controller on the lower-body to regulate balance [31, 140]. To control the upper limbs of the HRP-2, Peer *et al.* [140] adopt a common control scheme for position-controlled robots: the *admittance* controller. Such controllers define the reference through the differential equation of a virtual mass-spring-damping system:

$$\boldsymbol{f} = \boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{B}\dot{\boldsymbol{x}} + \boldsymbol{K}\boldsymbol{x} \tag{1.20}$$

where $\boldsymbol{x}$ is a reference pose in the task space, $\boldsymbol{M}$, $\boldsymbol{B}$ and $\boldsymbol{K}$ are arbitrary mass, damping and stiffness matrices of the equivalent virtual system, and $\boldsymbol{f}$ are wrenches exerted on the equivalent virtual point, i.e., the net wrenches measured at the contact point which are the references sent by the human operator. Under time delay, $\boldsymbol{M}$, $\boldsymbol{B}$, and $\boldsymbol{K}$ have to be selected appropriately to

guarantee stability of the overall teleoperation system, as done in [60] while teleoperating the HRP-2 located in Tsukuba (Japan) from Munich (Germany).

Whole-body bilateral teleoperation

An extension of the conventional idea of bilateral teleoperation is starting to be investigated to dynamically couple human and robot at a whole-body level [149]. This strategy consists of mapping the whole-body kinematic (joints position, velocitiy, etc) and dynamic (contact forces, joint torques, etc) references from humans to robots, while providing the operator with feedback regarding the robot's whole-body dynamics.However, the naturally unstable dynamics of humanoid robots poses an additional challenge to the whole-body teleoperation: the robot must balance while reproducing human movement.

The strategies for whole-body bilateral teleoperation utilize kinesthetic feedback to inform the operator about the robot's dynamics and stability in real-time. The strategy usually focuses on the center of mass dynamics, and other condensed information about the robot. For instance, the cable driven feedback interface in [143] exerts forces on the demonstrator's waist corresponding to the state of the robot's center of mass. This feedback allows the human to teach the robot how to compliantly interact with the environment. In [192], the feedback force applied to the human's torso is proportional to how close the robot is from tipping over. This is estimated by considering the distance between the robot center of pressure and the edge of the support polygon. The closer the robot is from tipping over in one direction, the larger the feedback force applied to the operator in the opposite direction. A similar strategy is used in [32] by providing discrete vibration levels to the operator using a belt with vibrotactile feedback. In [80], the force feedback device TABLIS, a powered exoskeleton, applies forces to the operator's feet to indicate that the robot is stepping onto an obstacle. This enables the operator to control the robot to navigate over objects. Finally, in [151] the force feedback is utilized to dynamically synchronize human and machine. The force feedback generates drag (negative feedback) if the robot cannot keep up with the operator movement, or it speeds human motion (positive feedback) if the robot moves faster than the operator. The high-level expression for the force feedback is given by:

$$\boldsymbol{f}_{fb} = k_H \left[\left(\dot{\boldsymbol{x}}'_R - \dot{\boldsymbol{x}}'_H \right) + \boldsymbol{f}'_{ext} \right], \tag{1.21}$$

where $\dot{\boldsymbol{x}}'_i$ is the dimensionless CoM velocity of the human (H) and robot (R) [145], $\boldsymbol{f}'_{ext}$ is dimensionless external forces applied to the robot, and k_H is a scaling factor proportional to the operator's size and body mass. This strategy enables human and robot to dynamically take simultaneous steps.

In general, in whole-body bilateral teleoperation, the kinesthetic feedback provided to the operator is proportional to some kinematic or dynamic discrepancy between human and robot in respect to the task at hand and/or the balance regulation. The shared control principle arises from the fact that the robot must prioritize between following human motion command to perform a task and maintain its own bipedal stability. Some strategies shift the balancing authority to the robot's autonomous controller. Which means that the robot is responsible for predicting if a given command will jeopardize stability and decide the best course of action to override the motion. For instance, in [79], the robot's controller uses stability considerations from the LIP model to modify the CoM trajectory commanded by the human, preventing the operator from destabilizing the robot. This represents a more conservative approach that guarantees stability of the movement of the robot, but prevents the operator from exploring motions that would go beyond the boundaries of the stability metric employed. In contrast, other approaches, such as [150], rely on the human operator to actively regulate the robot's balance. In this sense, the robot follows

human movement with only minor regulation from the robot's autonomous controller, and the operator must perceive the robot's destabilization through the kinesthetic feedback and mitigate it by adapting the commanded movement. Although riskier, this strategy frees the operator from abiding to the boundaries of predefined stability metrics, which are challenging to mathematically define in the context of legged robots. The goal of this approach is to eventually allow the operator to learn how to cope with the robot's dynamics and to create new motions on the fly.

Impedance control

A similar concept to admittance control can be employed in non-bilateral systems to provide the robot the ability to dexterously interact with the remote environment. In [31], Brygo *et al.* implement an autonomous *impedance* controller that regulates the robot's joints stiffness and damping according to the manipulation loading conditions. Also in this case a virtual mass-spring-damping system is employed, but the main difference between admittance control and impedance control is that the former controls motion after a force is measured, while the latter controls force after motion or deviation from a set point is measured [94]. In [31], the COMAN robot performs free-space motions with compliant limbs to ensure safe interactions during unforeseen collisions on the whole-arm. During the manipulation task, the controller stiffens the humanoid arm joints according to the the external force sensed at the end-effector, permitting to handle the task loads.

Alternatively, the impedance profile can be sent to the robot through a BMI applied to the master operator's arm, using for example non-intrusive position and electromyography (EMG) measurements, techinque also known as *tele-impedance* [21].

1.7 Summary and Discussion

We presented the general architecture of a complete teleoperation system. First, the user motion is captured by either using optical tracking systems, by recurring to inertial-based technology or by using an exoskeleton. In this case the exoskeleton is directly coupled with the robot and the system is said to be bilateral, since one device is used both as input and feedback. Once measured the motion is retargeted to the robot. Then the some robot's assistance can be used to modify the resulting motion trajectories of the robot, for example to compensate for delays or to help the user in performing a certain task better by recurring to shared control approaches. These approaches use some robot autonomy to improve the efficiency of the user teleoperating the robot. In fact there are certain tasks that might be difficult to perform, where small errors in motion inputs, lead to large errors in motion outputs, and some assistance from the robot can help the user accomplish that task. The resulting trajectories are then set as references for the controller of the robot, which generates the low level commands for the robot, which can be joint positions or torques. The robot and the environment states are measured, for example the forces acting on the robot or the images from the robot cameras are recorded and all the information is streamed back to the human operator through the communication channel as feedback. It can be sent as it is, like the sound from the robot site, or it can retargeted to the user, for example the forces measured at the robot side can be translated in vibration signals used to actuate a vibrotactile belt, to provide a sense of the robot's balance to the user.

In the literature there exists no such complete teleoperation system, due to complexity of implementing each composing element. The works presented in the literature generally focus on one or few components of the teleoperation system. Similarly in this thesis, we focused only on some components of the teleoperation system: first we measure the human motion, then we

translate it into an equivalent one for the robot; we use some robot's assistance to compensate for the delays in the network; and implemented a controller that makes the robot follow the computed reference trajectories. For the feedback, we stream the images of the robot's cameras and other external cameras to the user. These components have been implemented to tackle three main challenges regarding the whole-body teleoperation of humanoid robots:

- transferring the human motion to the robot so that the robot moves in a human-like way, despite differences in the body structure and in the dynamics;
- finding a controller that allows the robot to follow efficiently the human references while achieving balance. In the case of teleoperation this controller should be as generic as possible, since we do not know a priori the exact tasks that we will have to perform. This is different when we want to design specific autonomous robot behaviors which are typically custom tuned. Moreover, the controller testing is usually done in simulation and there is the possibility that working solutions in simulation cannot be transferred intuitively to the real robot;
- teleoperating the robot despite the delays. In space application these can get pretty significant. For example in the METERON project [107], where a robot on Earth was teleoperated from the ISS, typical delays were around 0.8s (with a maximum of 3s).

2

Motion Retargeting from Human to Robot

Transferring the motion from a human operator to a humanoid robot is a crucial step to enable robots to learn from and replicate human movements. The ability to retarget in real-time whole-body motions that are challenging for the humanoid balance is critical to enable humans to teleoperate humanoids. In this chapter of the thesis, we review existing retargeting methods and propose one that allows our humanoid robot iCub to replicate the motion of the human operator, acquired by a wearable motion capture suit, while maintaining the whole-body balance. We introduce some dynamic filter in the retargeting to forbid dangerous motions that can make the robot fall. We validate our retargeting approach through several experiments on the iCub robot, which has a significantly different body structure and size from those of the human operator. This work presents an original way to transfer the whole-body motion of the user to the robot and was published in [181]. We additionally report the results of experiments performed on a simulated HRP-4C robot, which has body dimensions more similar to those of the operator.

2.1 Retargeting Techniques

We define the retargeting process as *the mapping of human motion trajectories into references for the robot according to a scaling principle*, that is as a mapping $\mathbb{H} : A \rightarrow A'$ where A is the domain of the perceived human motions (kinematic and dynamic trajectories) and A' is the space of robot actions. In this definition, the scaling principle identifies the robot autonomy and the human authority in the teleoperation scenario. The mapping should be identified such that it minimizes the difference between the human intent and the robot actions while respecting the constraints in the teleoperation scenario. In practice, we can identify the retargeting as a cascade of mappings (Figure 2.1). In the higher level mapping $\mathbb{G} : S \rightarrow F$, the input are the user speech and the psychophysiological information S (Section 1.2.2), which determine the space of possible lower level retargeting F. $\mathbb{G}$ can be determined as a classification or regression problem. Two examples of the mapping $\mathbb{G}$ can be found in [54,170], where the authors developed attentive systems for real-time adaptation of the retargeting policy and the robot autonomy level. The robot can regulate its autonomy level according to the perceived user state and by estimating the confidence in the goal prediction and the task difficulty.

On the other hand, the retargeting $\mathbb{F} : A \rightarrow A'$ maps the captured human kinematics and dynamics information to reference trajectories for the robot. In the teleoperation literature, the term retargeting mainly refers to the mapping $\mathbb{F}$, while the mapping $\mathbb{G}$ is fixed offline by the

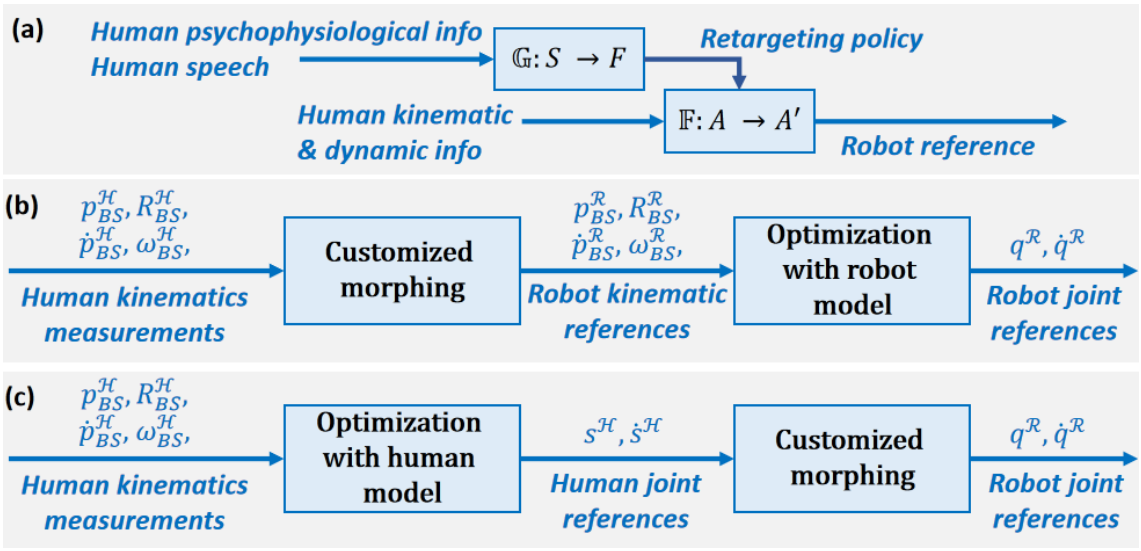


Figure 2.1: **Retargeting schemes.** a) the cascade of mappings for retargeting human actions and psychophysiological state (Section 1.2.2) to robot actions; b) kinematic retargeting in the task space; c) kinematic retargeting in the configuration space.

operator or by the developer of the system. In this thesis, only the retargeting $\mathbb{F}$ is considered.

There are several formulations for the retargeting $\mathbb{F}$ from human to robot. In literature, it is done either by using human or robot models, or based on learning methods at the kinematic level.

2.1.1 Model-based kinematic retargeting

In this method, the retargeting is done either in the task space or in the configuration space. In the task space retargeting, the Cartesian pose (or velocity) of one or more human body segments is mapped to corresponding values for the robot segments. Later, as shown in Figure 2.1, the inverse kinematic problem is solved by minimizing a cost function on the basis of the robot model while considering the robot constraints [43]. Different authors considered distinct body parts as the target of the mapping. A frequent approach is to map the motion of the human wrist to that of the robot end effectors [55, 79, 101, 192]. A commonly used mapping in the literature is to perform an identity map between the rotational motion of the human and the robot, which usually works because of the anthropomorphic robot design, whereas in the case of translational motion a fixed gain is used [55], to take into account the differences in size. A more complicated approach may identify this rotational and translational gain as a function of the human intention and ongoing task. To enhance the similarity of the robot motion to the human one, the retargeting of the elbow motion with a lower priority in the optimization problem is suggested in [106]. Alternatively in [43], similarity and scalability features are obtained by mapping all the rotational motions of the human limbs to the robot limbs with consideration of various robot constraints. In this work, the morphing policy is found manually offline. To overcome the manual morphing problem, Ayusawa *et al.* [27] suggested solving simultaneously the problems of geometric parameter identification of the morphing function and the inverse kinematics.

Configuration space retargeting refers to the mapping in the joint space from human to robot. This morphing is normally used when human and robot have similar joint orders. As demonstrated in Figure 2.1, the human measures and model are used in an inverse kinematics problem. The joint angles and velocity of the human joints are identified and mapped to the corresponding joints of the

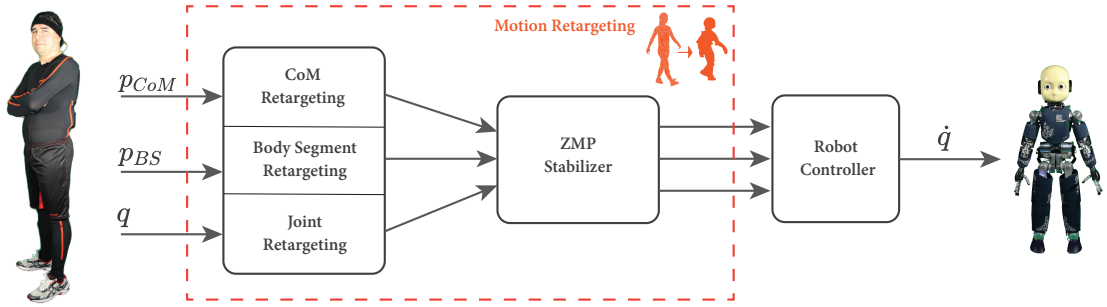


Figure 2.2: Teleoperation pipeline. In the motion retargeting block, the body segments pose, the joint angles and the center of mass are retargeted to the robot. The retargeted center of mass is corrected by a ZMP stabilizer. All the information is then fed to the robot controller that generates the velocity commands for the robot motors.

robot [106]. When the human joint ranges differs from those of the humanoid, the robot constraints should also apply in the morphing function. As explained later, our teleoperation system combines both retargeting strategies.

2.1.2 Learning-based kinematic retargeting

Different data-driven approaches for mapping the human motion to the robot one have been proposed in the literature. Although these methods are mainly used for generating animated character motions and do not encode the dynamics and balance of the robot, they can also be employed for humanoid robots retargeting. Pierce *et al.* [144] proposed a supervised learning approach to find the parameters of transformations in order to enable the robot to mimic the human motion with different similarity criteria. In the learning phase, the human mimics the robot motion on a 2D task space, while in the test phase the learned parameters are used for retargeting the human motion to the robot motion. In [99], a three-phase optimization problem based on reinforcement learning is presented. Human motions are extracted and mapped to the robot from a video stream. Later, the policy of the reinforcement learning problem is learned in order to maximize the similarity of the human and robot motion in the task space. Finally, an optimization problem is solved to minimize the error between the robot retargeted motion and the ground truth generated by the human demonstrations. Alternatively, Peng *et al.* [141] proposed a data-driven deep reinforcement learning framework is used for training control policies for simulated characters. Their method handles keyframed motions, highly-dynamic actions such as motion-captured flips and spins, and retargeted motions. By combining a motion-imitation objective with a task objective, characters were trained to react intelligently in interactive settings, e.g., by walking in a desired direction or throwing a ball at a user-specified target. A novel approach relying on unsupervised learning technique is implemented by Villegas *et al.* in [189], where recurrent neural networks and forward kinematics are blended. It relies on the cycle consistency principle, according to which motions retargeted to the avatar should generate the original motions of the human when retargeted back.

2.2 Methods

The first step in motion retargeting is to track the human pose through a motion capture system, that in our case was the Xsens system (Section 1.4). Then, the retargeting method translates the

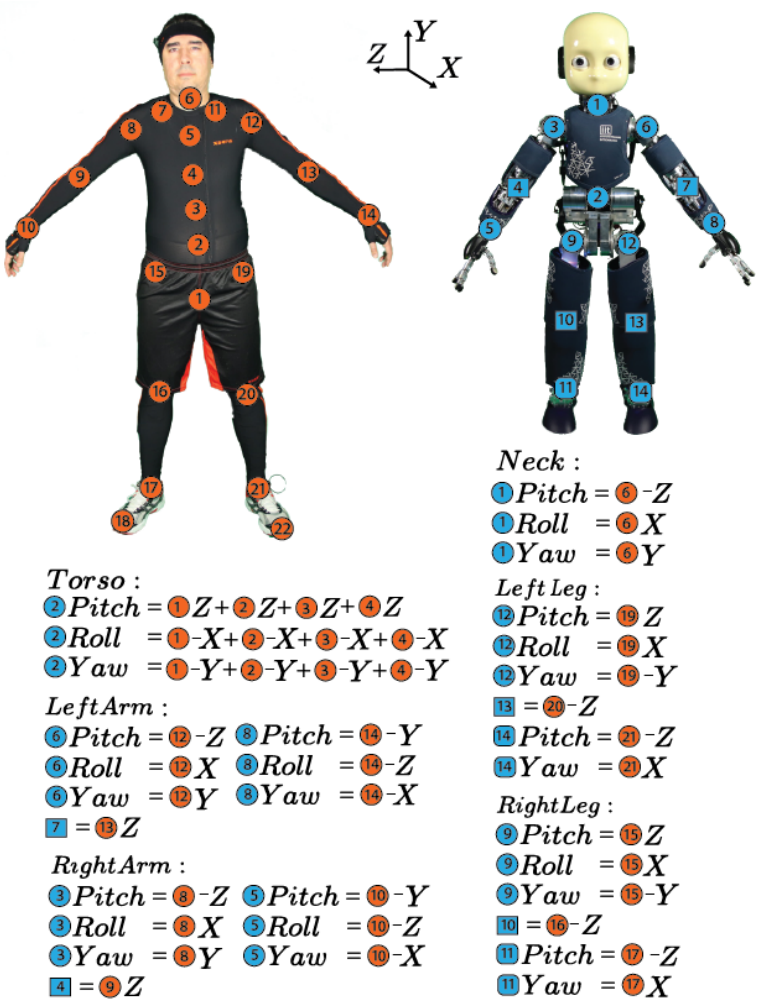


Figure 2.3: Posture retargeting from human (Xsens system) to iCub. Each joint of the robot is associated to a spherical joint of the Xsens skeleton together with its rotation axis.

captured human motion into equivalent reference trajectories for the robot controller, which has to track these reference values and compute the corresponding low-level commands for the robot's motors (Figure 3.2). This section first presents our retargeting strategy and then briefly describes the robot controller.

2.2.1 Joint angles mapping

This retargeting strategy was introduced in [181]. The Xsens model of the human has 66 DoFs that correspond to 22 spherical joints. Except for the torso, the other spherical joints can be easily visually assigned to the corresponding ones of the iCub. Less intuitive is the mapping between the individual joints composing the spherical one. Through several tests on simulation we identified the right mapping reported in Figure 2.3. For the torso, we used an approximate mapping. We considered the joints that are the most involved in the motion that are jL4L3, j1T12, and jT9T8, corresponding to the vertebrae going from the second lowest lumbar vertebra (L4) to the thoracic vertebra at the level of the breastbone (T8). The joints jT1C7 and jL5S1 (together with the other joints not reported in Figure 2.3) have been considered in some tests as well but

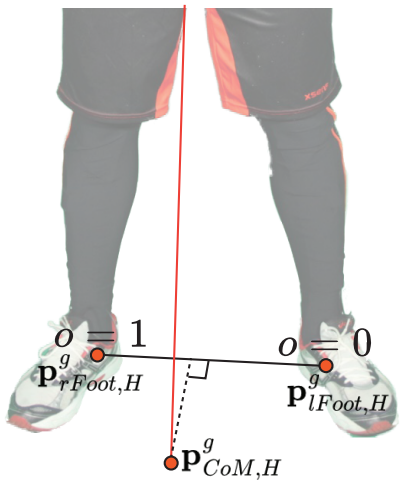


Figure 2.4: Normalized offset center of mass computation. The normalized offset is computed from the ground projection of the center of mass $\mathbf{p}^g_{CoM,H}$ and the feet positions of the human.

their contribution is negligible. Each joint value of the torso is obtained from the sum of the corresponding rotations of the joints jL4L3, j1T12, and jT9T8. Except for the yaw of the torso, this mapping is an approximation since the actual angle between the trunk and the hips is generally lower than the one given by the sum of the three joints. However, we noticed that the difference is not so relevant, that is reasonable if we consider that most of the inclination of the torso is generated by the hip movement.

After the identification of the human-iCub joints association, we consider the joint angle variations of the human with respect to the starting posture to compute the corresponding instantaneous values of the robot joint angles

$$\mathbf{q}^k_R = \mathbf{q}^0_R + (\mathbf{q}^k_H - \mathbf{q}^0_H), \tag{2.1}$$

where $\mathbf{q}$ is the joint positions vector, the superscripts 0 and k refer to measurements at initial time and at time k, and the subscripts H and R indicate measurements on human and robot, respectively.

2.2.2 Center of mass tracking

The center of mass of a distribution of mass in space is the unique point where the weighted relative position of the distributed mass sums to zero. Calculations in mechanics are often simplified when formulated with respect to the center of mass. It is a hypothetical point where the entire mass of an object may be assumed to be concentrated to visualise its motion. A person's center of mass is slightly below his/her belly button, which is nearly the geometric center of a person. Similarly, in humanoid robots, the center of mass is usually located around its waist.

To transfer the human center of mass, we use normalized offsets, from which we then reconstruct the robot center of mass ground position. We consider the ground projection of the human center of mass $\mathbf{p}^g_{CoM,H}$. Its position with respect to an arbitrary foot (let us say the left) is projected onto the line connecting the two feet. The result is then normalized to obtain an offset $o \in [0, 1]$

$$o = \frac{(\mathbf{p}^g_{CoM,H} - \mathbf{p}^g_{lFoot,H}) \cdot (\mathbf{p}^g_{rFoot,H} - \mathbf{p}^g_{lFoot,H})}{|| \mathbf{p}^g_{rFoot,H} - \mathbf{p}^g_{lFoot,H} ||^2}. \tag{2.2}$$

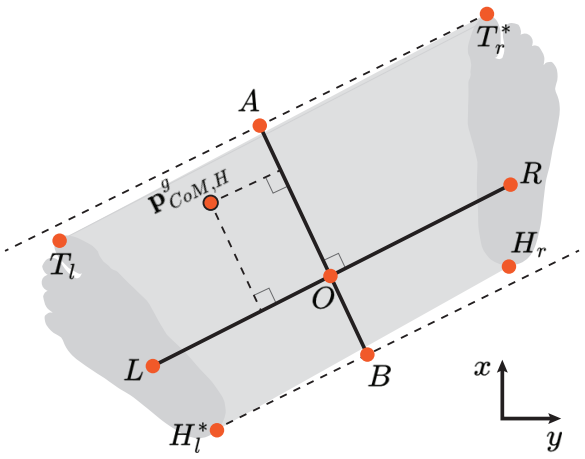


Figure 2.5: Determination of the normalized offset with respect to the line orthogonal to that connecting the center of the feet.

So when the human is in a symmetric pose in double support mode the offset has a value around 0.5 and when the human stands on a single foot the offset is either 0 (left foot) or 1 (right foot). The robot center of mass ground projection is then reconstructed on the line connecting the two feet by means of this offset value.

$$\mathbf{p}^{g,feetline}_{CoM,R} = \mathbf{p}^g_{lFoot,R} + o(\mathbf{p}^g_{rFoot,R} - \mathbf{p}^g_{lFoot,R}). \tag{2.3}$$

To retarget also changes of the human center of mass that are not on the line connecting the two feet, we first consider the toes and heels positions on the ground of the human, retrieved from the motion capture system. These positions represent points on the ground and we will call them T_l, T_r (toes) and H_l, H_r (heels). For simplicity, we will also refer to the feet positions on the ground as points L (left foot) and R (right foot). Moreover, we refer to the point in between L and R as O, and to the line on the ground connecting the feet as $\overline{LR}$ (Figure 2.5).

To determine which toe and which heel correspond to the outer edges of the support polygon (Figure 2.5), we have to calculate the distance d of each point T_i (H_i) and select that at the maximum distance from the line $\overline{LR}$, which will be indicated as T_i^* (H_i^*)

$$d(T_i^*, \overline{LR}) = max(d(T_l, \overline{LR}), d(T_r, \overline{LR})). \tag{2.4}$$

The line $\overline{LR}$ can be easily found from the points L and R. The line equation is $\overline{LF} : y = ax + b$ where

$$a = \frac{y_R - y_L}{x_R - x_L} \tag{2.5}$$

and the vertical intercept b is obtained by using the coordinate of R (or L) in the line equation:

$$b = -\frac{y_R - y_L}{x_R - x_L} x_R + y_R. \tag{2.6}$$

The next step is to find the equation of the lines parallel to the feet line $\overline{LR}$ and passing through the points T_i^* and H_i^*. These share the same slope a as $\overline{LR}$ and their vertical intercepts can be found by using in their equations the points T_i^* and H_i^* respectively. To determine the points A

and B on these lines (see Figure 2.5), we first compute the equation of the line $\overline{AB}$ orthogonal to $\overline{LR}$ and passing through O

$$\begin{aligned} \overline{AB} &: y = mx + q \\ m &= -1/a = -\frac{x_R - x_L}{y_R - y_L} \\ q &= \frac{x_R - x_L}{y_R - y_L} x_O + y_O. \end{aligned} \tag{2.7}$$

The points A and B can then be obtained as the intersection of $\overline{AB}$ with the lines parallel to $\overline{LR}$ passing through T_i^* and H_i^* respectively. A normalized offset o' can finally be computed in the same fashion as (2.2):

$$o' = \frac{(\mathbf{p}^g_{CoM,H} - \mathbf{p}_{B,H}) \cdot (\mathbf{p}_{A,H} - \mathbf{p}_{B,H})}{|| \mathbf{p}_{A,H} - \mathbf{p}_{B,H} ||^2} \tag{2.8}$$

From this, we can reconstruct the robot center of mass ground projection on the line $\overline{AB}$ of the robot, orthogonal to its feet line:

$$\mathbf{p}^{g,\overline{AB}}_{CoM,R} = \mathbf{p}_{B,R} + o'(\mathbf{p}_{A,R} - \mathbf{p}_{B,R}) \tag{2.9}$$

The final retargeted center of mass of the robot is obtained from $\mathbf{p}^{g,\overline{AB}}_{CoM,R}$, $\mathbf{p}^{g,feetline}_{CoM,R}$ and $\mathbf{p}_{O,R}$ as follows:

$$\mathbf{p}^g_{CoM,R} = \mathbf{p}^{g,\overline{AB}}_{CoM,R} + (\mathbf{p}^{g,feetline}_{CoM,R} - \mathbf{p}_{O,R}). \tag{2.10}$$

2.2.3 Retargeting body segment Cartesian poses

For transferring the translational movements of a body segment we use a fixed scaling factor

$$\alpha = \frac{l_R}{l_H} \tag{2.11}$$

given by the ratio between the length of the robot's limb associated to that body segment (e.g. the length of the arm, if considering the end-effector position) and that of the human when in N-pose[3]. As for the joint angles, we considered the deviation of the end-effector positions from their starting value over time. The instantaneous reference value of the robot is then computed as:

$$\mathbf{p}^k_{BS,R} = \mathbf{p}^0_{BS,R} + \alpha(\mathbf{p}^k_{BS,H} - \mathbf{p}^0_{BS,H}), \tag{2.12}$$

where $\mathbf{p}_{BS}$ is the position vector of a body segment (e.g. an end-effector), the superscripts 0 and k refer to measurements at initial time and at time k, and the subscripts H and R indicate measurements on human and robot, respectively. In the experiments with iCub reported in this chapter, only the height of the waist of the human was retargeted to the robot, with α given by the ratio of the waist height of the robot and that of the human when in N-pose. In other experiments performed in the next chapters also the Cartesian positions of the end-effectors was taken into consideration.

To retarget the orientation of a body segment, we use an identity map between the rotational motion of the human and the robot. As done when retargeting the position of a body segment, we consider only the variation of rotation with respect to the initial orientation as follows:

$$\mathbf{R}^k_{BS,R} = \mathbf{R}^0_{BS,R} \mathbf{R}^k_{BS,H} (\mathbf{R}^0_{BS,H})^{-1}. \tag{2.13}$$

[3]N pose is a resting pose, where the human stands with lowered arms close to its body.

2.2.4 ZMP retargeting correction

During whole body teleoperation of humanoid robots, disastrous crashes may occur if the desired center of mass trajectories recorded from the human do not ensure the balance of the controlled robot when retargeted. To this scope, we propose a QP-based "preprocessor" that adjusts in real-time the desired center of mass to satisfy constraints that represent a condition for dynamic balance. A very common strategy relies on the concept of Zero Moment Point (ZMP) [57], which is the point with respect to which the tipping moment generated by the gravity and the inertial forces is zero. If this is kept at all times inside the support polygon of the robot, the motion is dynamically feasible. The dynamics of the ZMP can be derived by moment balance. By neglecting rotational contributions and assuming a constant height of the center of mass, the dynamics can be rendered linear. This model can be interpreted as a Linear Inverted Pendulum (LIP) [90].

Through the LIP model it is possible to establish a simple relation between the ZMP and the center of mass dynamics. By neglecting rotational terms and assuming a constant height h_{CoM} for the center of mass, the moment balance equation of the robot leads to:

$$x_{ZMP} = x_{CoM} - \frac{1}{\eta^2}\ddot{x}_{CoM} \tag{2.14}$$

where $\eta = \sqrt{g/h_{CoM}}$, with g the gravitational constant, while x_{CoM} and x_{ZMP} are respectively the CoM and ZMP positions along x (similarly for y).

By employing the dynamic equation of the LIP (2.14), it is possible to set up a QP optimization problem that provides at each control iteration a correction of the desired center of mass that satisfies the balance condition on the humanoid

$$\begin{aligned} &\min_{x_{ZMP}} (\dot{x}^{ref}_{CoM} - \dot{x}_{CoM})^2 \\ \text{subject to: } &\dot{x}_{CoM} = \dot{x}^{m}_{CoM} + \tfrac{\delta g}{h^{m}_{CoM}}(x_{CoM} - x_{ZMP}) \\ &x^{min}_{ZMP} < x_{ZMP} < x^{max}_{ZMP} \end{aligned} \tag{2.15}$$

where $\dot{x}^{ref}_{CoM}$ is the reference human retargeted center of mass velocity, δ is the sampling time, $\dot{x}^{m}_{CoM}, h^{m}_{CoM}$ are respectively the last center of mass and the last center of mass height measured from the robot, x^{min}_{ZMP} and x^{max}_{ZMP} are the lower and upper bound of the support polygon of the robot and the first constraint is derived from (2.14) using the Euler approximation.

2.2.5 Robot controller

Once the data is mapped to feasible corresponding values for the robot, these are set as references for the controller (see Figure 3.2). We opted for a velocity-based Quadratic Program (QP) controller based on OpenSoT [75], an open-source library implementing QP controllers based on the *stack-of-tasks* [74]. Nowadays, QP controllers have become widespread thanks to their flexibility and various formulations have been proposed in the literature, [62, 131, 132], which take into account joint velocity, acceleration and torque control as well as contact forces. They enable to handle various type of constraints that, depending on the chosen formulation, may include robot dynamics, friction cones, self-collision avoidance, joint limits and many more [132].

For the teleoperation, this kind of controllers is appropriate since they allow us to take into account both Cartesian tasks (body segment positions) and postural tasks while satisfying all the robot constraints. Moreover, the tasks can be specified both in a hard and soft priority fashion. More details about the literature of robot controllers used for teleoperation are given in Section

1.5. In the next chapter we show how we obtained an optimal controller that tracks efficiently the retargeted information, whereas the results reported in this chapter (which is mainly focused on the motion retargeting problem) were obtained using a first hand-tuned controller.

The multitask QP controller allows us to define Cartesian tasks and a postural task together with their subtasks. More specifically, we did not consider a single postural tasks but we separated it in several subtasks: $neck_q$ (neck joints), $torso_q$ (torso joints), $larm_q$ (left arm joints), $rarm_q$ (right arm joints).

We additionally considered a Cartesian task for the ground position of the center of mass of the robot (com_g) and a Cartesian task for the height of the floating base ($base_h$). The global position of the feet was also taken into consideration, in order to keep each foot in contact with the ground whenever it is a support link ($lfoot$ and/or $rfoot$). The posture of the legs was retargeted indirectly through the center of mass and floating base tasks. Our resulting selected stack of tasks for the controller was the following:

$$stack = (lfoot + rfoot + neck_q)/ \\ (com_g + base_h + torso_q + larm_q + rarm_q);$$

where $T_a + T_b$ means that the tasks T_a and T_b are in a *soft (Weighted) priority* relation while T_a/T_b means the tasks T_a and T_b are in a *hard (Null-Space) priority* relation. In single support mode, the task of the foot that is not the support one can be removed and a leg postural task can added for the lifted leg. However, in this work we only show experiments in double support. Given the considered tasks, the controller solves a QP program that is equivalent to an IK problem, generating velocity commands for the robot. As constraints, here we considered only joint position and velocity limits.

2.3 Results

2.3.1 Experiments with the iCub robot

We set up two experiments to validate our approach. First, through dynamics simulations on the simulated robot in Gazebo, we showed how the ZMP retargeting correction is essential to retarget a motion that is dynamically stable onto the robot. Then, we tested our framework by teleoperating the real robot iCub in real-time. Videos of our experiments with the real robot can be seen at `https://youtu.be/iZVAacyvYhM`.

Simulated robot – We selected three kind of motions to show the efficacy of our method: squat motion, hip roll exaggerate motion and a grasping motion involving some torso and leg movements (see Figure 2.8). Figure 2.6 shows how the ZMP associated to the retargeted motion without the correction lies outside the support polygon, making the robot fall inevitably (that is why we show this in simulation). This is due both to mechanical limitations of the robot, which cannot achieve the same center of mass displacement given the retargeted joint values (that might go beyond the robot joint limits), and to the generation of some momentum different from that of the human, which makes the desired robot center of mass trajectory unstable. Hence, the ZMP trajectory is corrected in real-time to stay inside the support polygon and the center of mass trajectory is modified accordingly (see Figure 2.7).

Real robot – We teleoperated the robot trying to move all its links to show the effectiveness of our framework. During the teleoperation the ZMP position of the robot always lies inside the support polygon as expected (see Figure 2.9). The robot joint trajectories follow the retargeted values guaranteeing the mimicry of the human motion (see Figure 2.10-2.11). The joint angles of

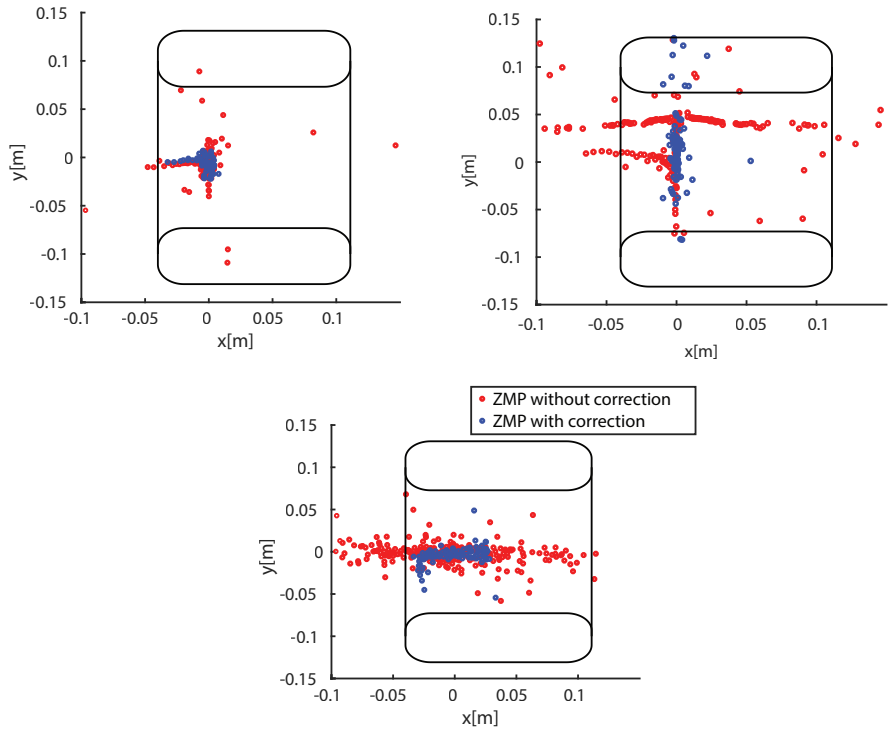


Figure 2.6: Robot ZMP position with and without dynamic correction. ZMP position of the simulated robot without (red) and with (blue) the stability correction during the teleoperation while performing a squat motion (top-left), a hip roll exaggerate motion (top-right), a grasping motion involving some torso and leg movements (bottom).

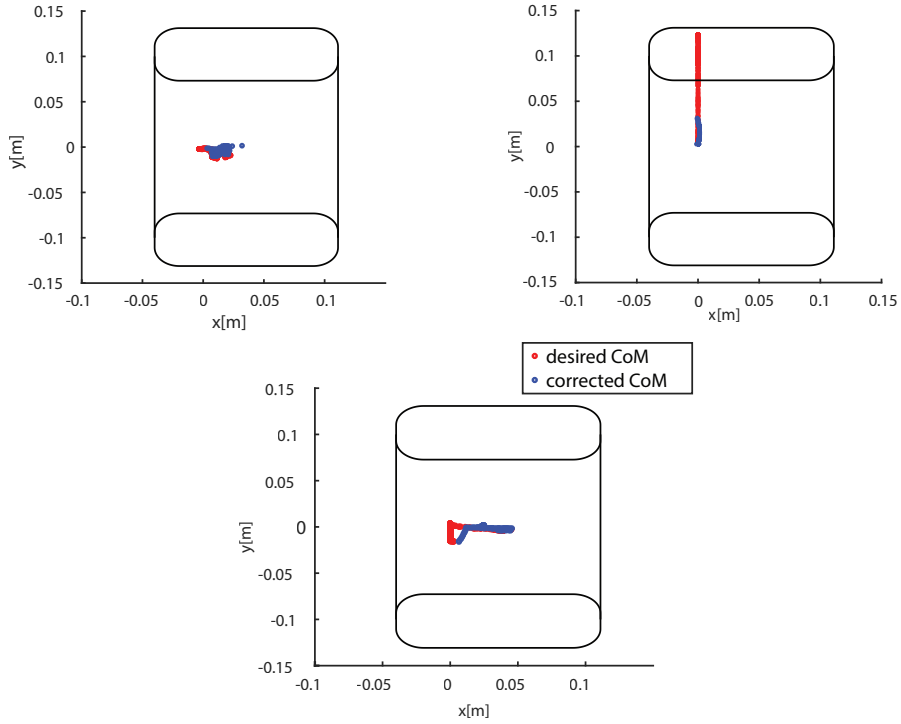


Figure 2.7: Robot center of mass position with and without dynamic correction. Center of mass desired (red) and corrected (blue) position of the simulated robot during the teleoperation while performing a squat motion (top-left), a hip roll exaggerate motion (top-right), a grasping motion involving some torso and leg movements (bottom).

Figure 2.8: Snapshots of simulations of the teleoperated robot and of the Xsens skeleton while performing a squat motion, a hip roll exaggerate motion and a grasping motion.

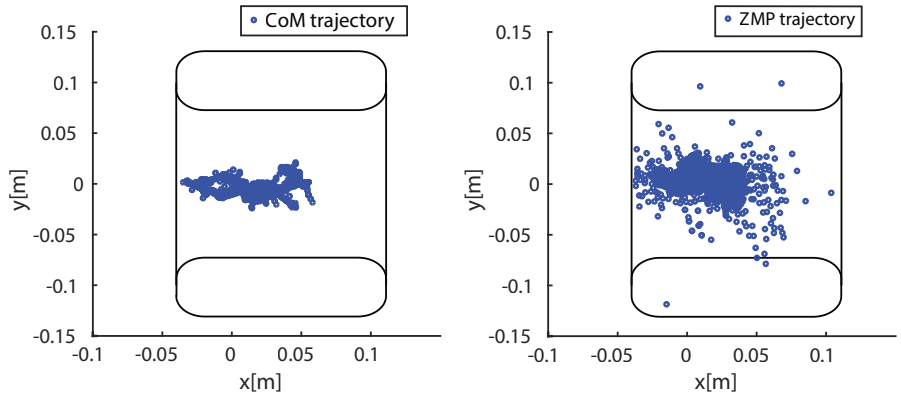


Figure 2.9: Center of mass (left) and ZMP (right) trajectories of the robot iCub during the teleoperation experiment.

Figure 2.10: Snapshots from the teleoperation experiment. The robot does not fall when the human operator falls back (frame on the bottom-right).

the legs are not taken into consideration in the stack, and the robot is free to adapt the posture of the legs according to the retargeted center of mass and waist height. Hence, the resulting trajectories of legs do not follow precisely the human reference (Figure 2.11), except for the knees and of the pitch of the ankles, which are close to the those of the human because of the waist height tracking. Even if the human performs some motions that are very challenging for the balance, the ZMP constraints introduced by our framework make the robot maintain its balance. The video attachment at `https://youtu.be/iZVAacyvYhM` shows that even if the operator is falling, the robot does not and keeps its balance.

2.3.2 Additional experiments with the HRP-4C robot

We performed additional experiments with the humanoid robot HRP-4C (Section 1.4.4) in simulation to test an alternative methodology to retarget even more human-like robot motions. In this case we considered the same tasks as the previous experiments (see section 2.2.5), but with the addition of a postural task for the legs. The inclusion of this task is essential to achieve a more

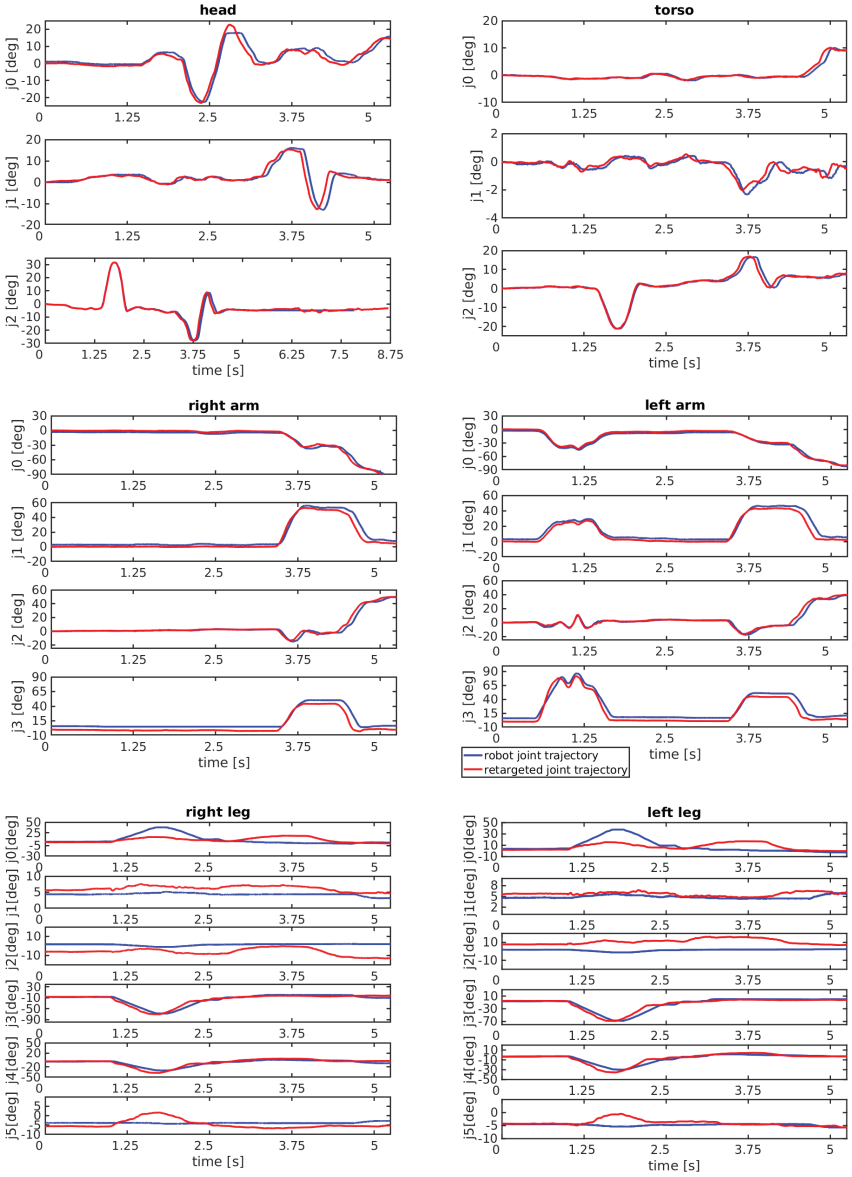


Figure 2.11: Joint trajectories of the robot iCub (blue) compared to the retargeted values (red) during part of the teleoperation experiment.

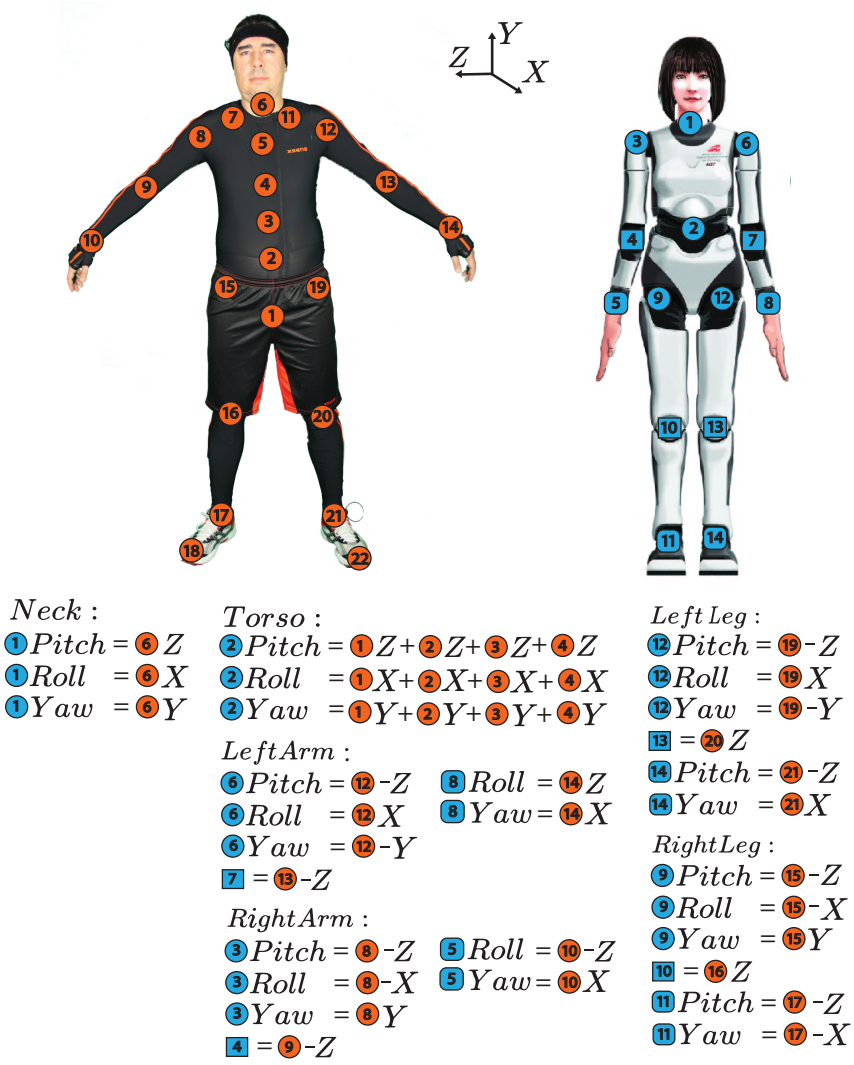


Figure 2.12: Posture retargeting from human (Xsens system) to HRP-4C. Each joint of the robot is associated to a spherical joint of the Xsens skeleton together with its rotation axis.

human-like motion since most of the trunk motion in humans is generated from the hip joints. Also in this case we identified a mapping between the joint angles of the human and the corresponding ones of the robot (Figure 2.12) after several testing in simulation.

Differently from previous experiments, we relied on a task space inverse dynamics (TSID)-based QP controller implemented in the framework mc_rtc [8]. This time the constraints not only bounded the control input, but also included the dynamics of the robot, ZMP constraints and environmental factors, such as friction cone constraints at the contacts. This allowed us to not use the ZMP-based correction of the retargeted center of mass information and to also consider as reference the posture of the legs retargeted from the human. Both are corrected internally by the controller to guarantee the contact stability and, more in general, to satisfy the constraints of the control problem. We used the following stack:

$$stack = (lfoot + rfoot + 0.1neck_q + com_g + \\ + 0.1torso_q + 0.1larm_{sub} + 0.1rarm_q + 0.1lleg_q + 0.1rleg_q);$$

Figure 2.13: Snapshots of the teleoperation of the HRP4C robot in simulation with the corresponding Xsens human skeleton input.

We tested the system on a ~100second-long sequence involving a variety of whole-body motions (see Figure 2.13). As emerges from Figure 2.14, this time the posture of the legs of the robot follows more precisely the retargeted values even when performing movements that solicit a lot the lower limbs. The corresponding video is avalaible at mybox.inria.fr/f/d48d78611730425bb8a6/?dl=1. The waist height is not retargeted anymore since the imitation of the human posture makes it possible to achieve indirectly a waist pose of the robot similar to that of the human. The contact constraints also allow the robot to bend the torso more by using the rotation of the hip joints without breaking the contacts.

2.4 Discussion

We proposed a retargeting method to transfer the motion from a human operator wearing a motion-capture suit to a humanoid robot. The method allows a real-time retargeting of generic double support motions. The generality is obtained thanks to the mapping of all the main human joints: not only the arms and the legs, but also the torso and the head. The center of mass as well is retargeted in a way that does not restrict the range of possible retargeted robot motions. The robustness is given by a ZMP correction approach that guarantees the dynamical feasibility of the retargeted trajectory in double support. These references have been fed to a controller based on an IK control scheme with a QP optimization solver, which generates the low-level commands for the robot.

The limitation of the approach is that it relies on a correction of the center of mass, which is based on a simplified robot model (the LIP). For motions that involve dynamic motions of the upper body, the LIP assumption would not be valid anymore and the correction we proposed in this chapter could generate center of mass trajectories that are not reliable. Also for locomotion on uneven terrains or motions involving multiple change of contacts, the LIP model would not be appropriate and more complex models should be used [111].

We also tested a different control scheme where the QP optimization solves an ID problem with constraints regarding the dynamics of the robot, its balance and contact conditions. With this

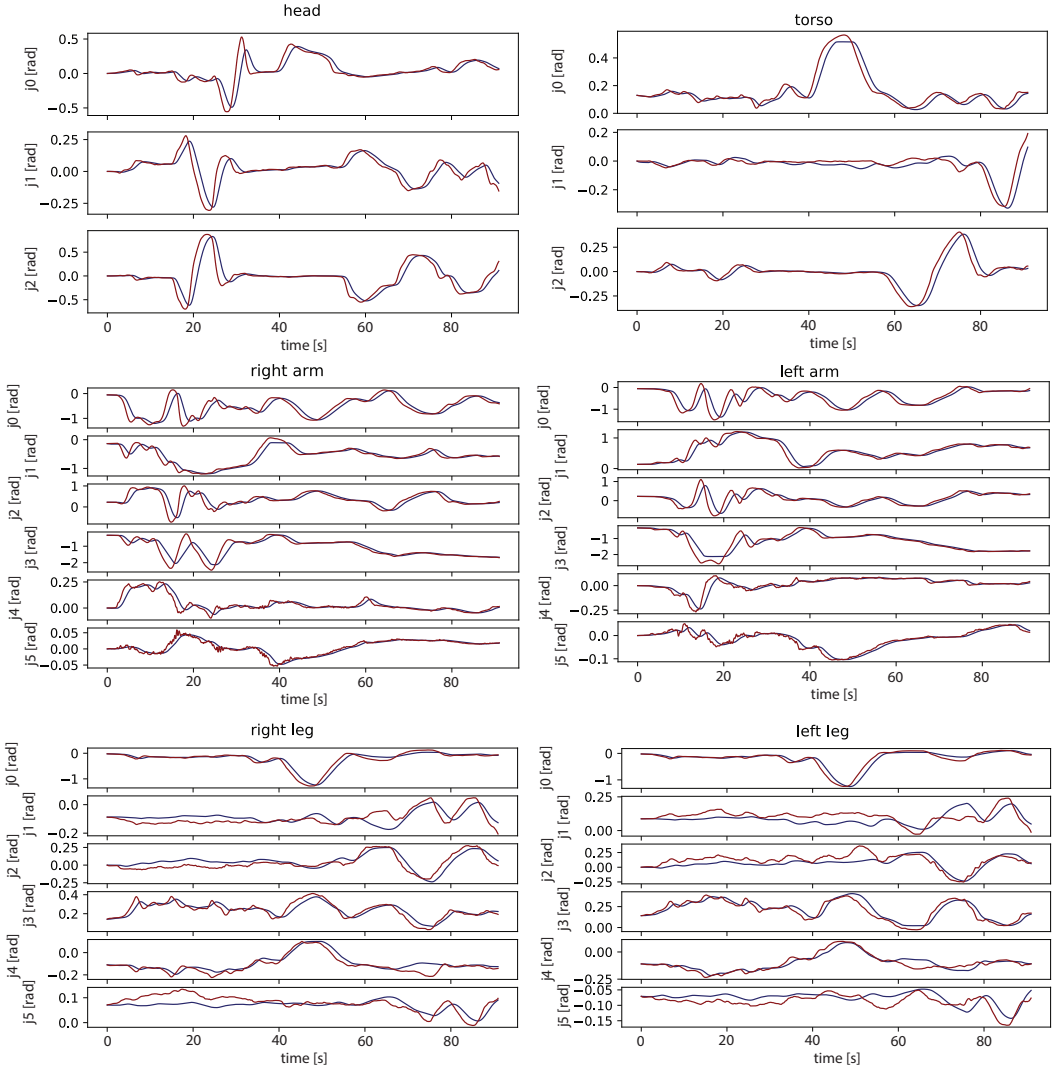


Figure 2.14: Joint trajectories of the robot HRP-4C (blue) compared to the retargeted values (red) during part of the teleoperation experiment in simulation.

formulation we achieved more human-likeness in the robot motion, especially at the level of the lower body. Moreover, with this control scheme we did not require any correction of the retargeted center of mass, which was performed internally by the controller in order to maintain the robot balance.

In the future we will extend the retargeting to more dynamic motions. In this case, since the human-like center of mass reference trajectories could be dynamically unstable for the robot the possibility that the control solver could not be able to find a feasible solution that respects the robot constraints is not to exclude. A solution could be to merge the two methods we investigated, that is both correcting the references retargeted from the human and relying on a controller that guarantees the satisfaction of the constraints of the robot and its environment. Although for periodic dynamic motions like walking, a better solution could be to use a higher-level retargeting strategy, as we explain in Chapter 4, where the robot is guided by the human while performing autonomously or semi-autonomously a given motion. This clearly requires that the robot is able and has been previously programmed to perform that motion autonomously.

3

Learning Optimal Controllers for Humanoid Robots

When controlling the whole-body of redundant robots like humanoids, several tasks have to be taken into consideration simultaneously. For example, the center of mass of the robot has to be controlled in a way that guarantees the robot's balance, while the position of a hand has to follow a specific trajectory to reach a certain object, the head has face toward that object and the torso has to bend toward that object. How reactively each of these tasks is tracked by the controller and with what priority, determines the final robot behavior. Finding the right task priorities, together with all the parameters that affect the controller performance, is often a time-consuming trial-and-error tuning procedure that requires human expertise and many tests on the robot. The procedure is even more demanding in a teleoperation scenario where we do not know exactly a priori what motion the robot is going to perform and where we need a generic and robust solution.

In this chapter, we investigate the possibility to automatically optimize off-line the parameters and priorities for generic multi-task controllers, so to obtain an optimal controller for teleoperation that tracks efficiently the retargeted human references. We first review the different approaches that have been proposed in the literature to optimize of some of the controller's parameters (Section 3.1). Then we present our optimization approach (Section 3.2) and we validate it on the robot iCub through several experiments (Section 3.3). The work presented in this chapter was published in [182].

3.1 Related Work and Contributions

In the literature, several approaches have been proposed to find optimal control configurations, the majority of which has been focused on learning soft priorities. In [124] for example, the authors parameterize the weight of each task by radial basis functions defined over time, allowing a temporal adaptation of the different soft priorities to the specific motions. A derivative-free stochastic optimization is used to learn the optimal parameters. Lober *et al.* [112] propose a framework that modifies a set of initially interfering tasks, via stochastic optimization of their parameters. They use Gaussian kernels to compute variance-dependent weights, which make it possible to handle conflicts between tasks. Paraschos *et al.* [136] use probabilistic movement primitives to learn the accuracy of different tasks. They exploit the variability of demonstrations given in Cartesian and joint spaces as a prioritization criteria to organize the different tasks. In [46], Dehio *et al.* use a mixture of controllers for whole-body motion generation and considered the mixture coefficients as policy parameters, optimizing them by means of a derivative-free stochastic

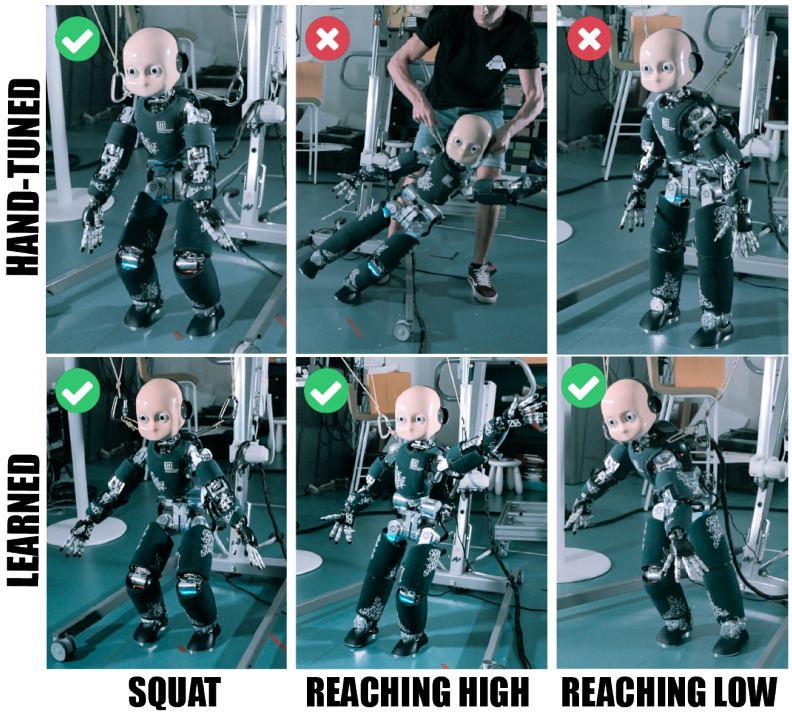


Figure 3.1: Hand-tuned vs learned controller. Despite being tuned to perform several motions in simulation, a hand-tuned controller can easily fail when transferred on the real humanoid and when performing different tasks that challenge its balance. With our multi-objective optimization approach, the learned controller can work on the real robot iCub, also on different test motions.

optimization algorithm. Their approach enables the transfer of the learned policy to new tasks, but the optimized mixture coefficients are not verified on the real robot.

For strict hierarchies, solutions have been proposed to overcome discontinuities in the control problem [95,98], arising when priorities are continuously rearranged. Dehio and Steil [47] propose a dynamically-consistent generalized hierarchical control that allows choosing, for each pair of tasks, between a soft or a hard priority with one task having null effect on the other one. However, their method is only validated in simulation on a manipulator.

Another combination of soft weighting and hierarchies is proposed in [169], where Silvério *et al.* introduce an identification of demonstrated priority behaviors, given an initial set of candidate task hierarchies, that allows the robot to reproduce the learned priorities in new situations. The demonstrations are given by implementing an inverse kinematics controller with the desired hierarchies in the simulated robot and making it track moving references. Both [47] and [169] limit the number of tasks to three.

Overall, the previous works propose interesting solutions for learning simultaneously soft and hard priorities, but for few tasks and in simulation. This is not enough for the whole-body control of real humanoid robots, where we usually have a higher number of tasks to fulfill. On top of that, learning solutions in simulations does not guarantee that such solutions will work on the real robot: this is a known problem caused by the *reality gap*.

Achieving transferable solutions is the central focus of many recent works in robotics, [24], [185], [40]. For example, optimal control and movement primitives are combined in [40] to find solutions which can be easily deployed on the real robot. However, they strongly rely on the accu-

racy of the simulated model to ensure the transferability of solutions. Another common approach that can help achieving the adaptability of the learned solution on new scenarios is Domain Randomization (DR) [185], which consists in randomizing some aspects of the simulation to enrich the range of possible environments experienced by the learner. In [24], robust policies for pivoting a tool held in the robot's gripper are learned in simulation, given random friction and control delays, such that the learned policies prove to be effective on the real robot as well. In [123], the learning procedure guarantees strict constraints fulfillment, but transferring knowledge from simulation to reality does not fully achieve the desired behavior. This implies that constraints satisfaction can be beneficial, but not enough to achieve transferability. Indeed to achieve a better generalization, solutions which are robust rather than optimal are needed. For instance, Del Prete *et al.* [48] proposed to improve the robustness of task space inverse dynamics by modeling uncertainties in the joint torques of humanoid robots. Charbonneau *et al.* instead, learned the soft priorities that optimize for both task performance and robustness [35], by applying the idea of DR. However, both [48] and [35] lack experimental validation on the real robot.

Differently from previous work, in this chapter we propose to learn both the structure of the controller (i.e., the position of each task in the hierarchy) and all its parameters (i.e., the task weights and Feedback Gains). We optimize for a large number of tasks and introduce a parametrization of the strict priorities that allows us to optimize them as simply as the soft weights. We look for solutions that are both high-performing in executing desired trajectories (low tracking error) and "robust" (reducing the tipping moment of the robot).

Any state-of-the-art motion generator [34] still synthesizes dynamically balanced trajectories for inaccurate robot models, without guarantees that a perfect tracking of these desired trajectories is possible on the real robot. At the same time, reactive whole-body controllers have often a very conservative design in terms of balancing, which prevents the robots to track desired trajectories that are instantaneously challenging for balance. It seems appropriate to reason in terms of compromise between tracking performance and balancing/robustness to look for controllers that can effectively work on real humanoid robots.

Instead of arbitrarily combining the two objectives (performance and robustness) into a single cost function, we follow a multi-objective optimization approach: we seek to obtain in a single optimization run the set of the Pareto-optimal solutions, i.e., the optimal trade-offs between the objectives. This enables the robot user to test candidate controllers from the Pareto front directly on the robot, without requiring further optimization or manual tuning. The proposed method is detailed in Section 3.2.

To summarize, we address the problem of automated learning of controllers for redundant robots with the following contributions: first, we introduce a parametrization of strict task priorities, so to learn simultaneously the hierarchical structure and the parameters of the controller with a multitude of concurrent tasks; second, we use multi-objective optimization algorithm to get a set of Pareto-optimal solutions (controller configurations) to be tested by the user on the real robot.

We validate our optimization process on the humanoid robot iCub, for double support motions. We compare controllers that were automatically optimized on a training sequence of motions against a baseline hand-tuned controller, over different motion sequences (see Figure 3.1). We further show that our optimized controller can be successfully employed to execute a variety of motions, other than those used in the training, for example for teleoperation, where reference trajectories (unknown a priori) are generated in real-time by the human operator.

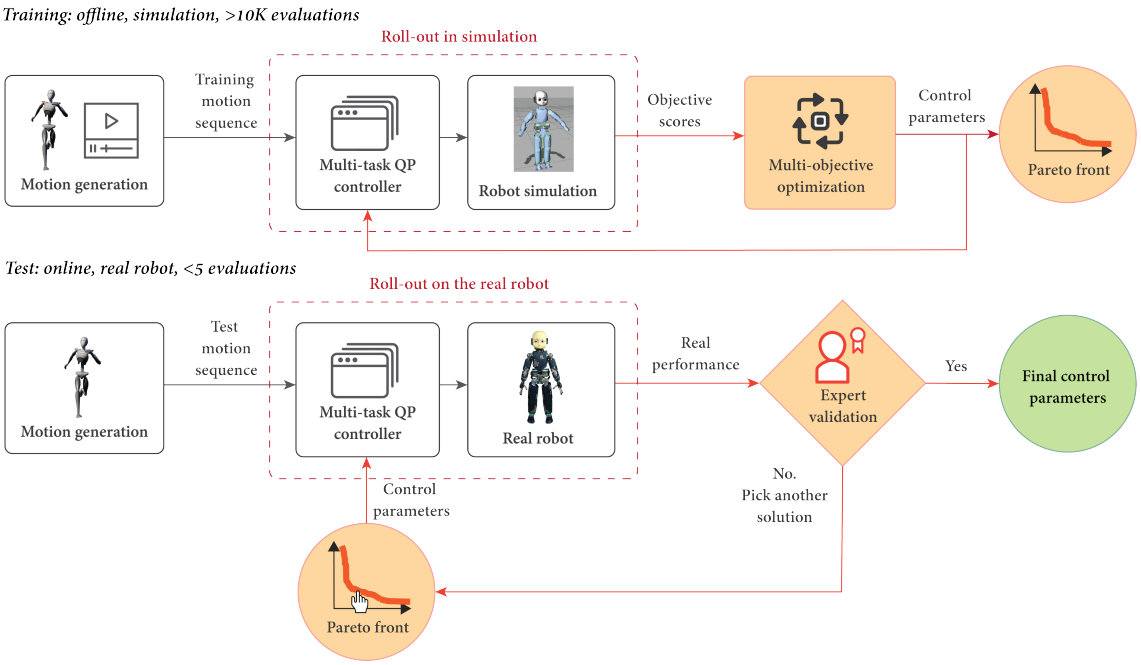


Figure 3.2: Overview of the approach for learning optimal controllers. Top: Optimization process (offline). A training motion sequence is generated. The optimization algorithm searches for the best controller configurations that make the simulated robot execute the reference motion sequence. The algorithm computes a set of Pareto-optimal control configurations, i.e. optimal trade-offs between robustness and performance. Bottom: Testing process (online). The user selects the most appropriate control configuration from the Pareto-optimal solutions for the real robot, getting a valid working solution in few trials.

3.2 Methods

The goal of our method is to automatically learn a control configuration of a controller enabling the real robot to execute a variety of different reference motions (for a given category of movements, e.g. double support). We define as **"control configuration"** of a controller with n tasks $\{\mathcal{T}_1, \ldots, \mathcal{T}_n\}$, the set of the Soft Priorities Weights w_k, the task feedback gain λ_k and the strict priority relations between the tasks, formalized as the level of each task $\mathcal{T}_k$ in their stack $\mathcal{S}$ (Section 4.3.1).

We look for a "generic" control configuration that trades-off performance on several motions, rather than a "motion-specific" controller that is highly optimized for a given movement but requires the optimization to be re-run for every other motion. We also want the result of the optimization procedure to be transferable onto the real robot.

Figure 3.2 is a flowchart of the learning procedure. First, a multi-objective optimization process based on roll-outs performed on a simulated robot model computes off-line a set of Pareto-optimal solutions. Each solution is a control configuration that trades-off different performance criteria (i.e., tracking score and a robustness/balancing score), optimized on a training sequence. Then the user selects candidate solutions from the Pareto front and validates them on the real robot using test motion sequences. Here, the purpose is to find transferable solutions that are high-performing and well-balanced / "robust", i.e., reducing the tipping moment of the robot. In the following, we detail all the steps of the proposed method.

3.2.1 Motion Generation and Task Specification

The Motion Generator is a module generating a sequence of robot movements, defined by reference trajectories $\boldsymbol{b}_k$. Any motion planning algorithm could be used. Here, without losing generality, we use our motion retargeting framework from Chapter 2, which allows us to generate references for the End Effectors (EEs) both in the Cartesian and postural space from the imitation of human movements.

The tasks $\mathcal{T}_k$ considered for the controller are those for which the motion generator produces the references $\boldsymbol{b}_k$.

3.2.2 Control Configuration – parameters to optimize

For each task $\mathcal{T}_k$, we want to learn its Soft Priority Weight (SPW) w_k, its Feedback Gains (FG) (position error gain λ_k and orientation error gain σ_k for Cartesian tasks, postural error gain μ_k for joint tasks) and its position in the stack/hierarchy. A specific task $\mathcal{T}_k$ can be either active in one level $\mathcal{S}_i$ of the stack $\mathcal{S}$ or completely deactivated (i.e., not in the stack). The parameter l_k called Hierarchy Level Selector (HLS) encodes the activation of the k-th task in $\mathcal{S}$ as follows[4]:

$$\begin{cases} \mathcal{T}_k \subseteq \mathcal{S}_1 & \text{if } (0 \leq l_k \leq 0.25) \\ \mathcal{T}_k \subseteq \mathcal{S}_2 & \text{if } (0.25 < l_k \leq 0.5) \\ \mathcal{T}_k \subseteq \mathcal{S}_3 & \text{if } (0.5 < l_k \leq 0.75) \\ \mathcal{T}_k \; deactivated & \text{if } (0.75 < l_k \leq 1) \end{cases} \tag{3.1}$$

Since humanoid robots exhibit bilateral symmetry, we used the same values for the weights w_k and the activation levels l_k for tasks related to the left and right parts of the robot. The control parameters all consist of real values $\in [0, 1]$.

[4]We consider only three levels of priorities, which is a reasonable assumption in humanoids' controllers, especially when dealing with a large number of tasks (TABLE 3.1). However, one could optimize for as many levels as they wish.

Table 3.1: Considered tasks, associated symbols and control parameters (Soft Priority Weight (SPW), Hierarchy Level Selector (HLS) and feedback gain (FG)) for the controllers C1 and C2.

Task		**Control Parameters**		
Description	Symbol	SPW	HLS	FG
left foot pose	$\mathcal{T}_{lf}$	w_f	l_f	$\lambda_{feet}, \sigma_{feet}$
right foot pose	$\mathcal{T}_{rf}$	w_f	l_f	$\lambda_{feet}, \sigma_{feet}$
left hand position	$\mathcal{T}_{lh}$	w_{ha}	l_{ha}	λ_{hand}
right hand position	$\mathcal{T}_{rh}$	w_{ha}	l_{ha}	λ_{hand}
com height	$\mathcal{T}_{cz}$	w_{cz}	l_{cz}	λ_{com}
com (x,y)	$\mathcal{T}_{cxy}$	w_{cxy}	l_{cxy}	λ_{com}
waist height	$\mathcal{T}_{wh}$	w_{wh}	l_{wh}	λ_{waist}
head orientation	$\mathcal{T}_{h}$	w_h	l_h	σ_{head}
chest orientation	$\mathcal{T}_{c}$	w_c	l_c	σ_{chest}
neck posture	$\mathcal{T}_{n}$	w_n	l_n	$\mu_{posture}$
torso posture	$\mathcal{T}_{t}$	w_t	l_t	$\mu_{posture}$
left arm posture	$\mathcal{T}_{la}$	w_a	l_a	$\mu_{posture}$
right arm posture	$\mathcal{T}_{ra}$	w_a	l_a	$\mu_{posture}$
left lower arm posture	$\mathcal{T}_{lla}$	w_{la}	l_{la}	$\mu_{posture}$
right lower arm posture	$\mathcal{T}_{rla}$	w_{la}	l_{la}	$\mu_{posture}$
left leg posture	$\mathcal{T}_{ll}$	w_l	l_l	$\mu_{posture}$
right leg posture	$\mathcal{T}_{rl}$	w_l	l_l	$\mu_{posture}$

Controller	**Considered tasks**
C1	$\mathcal{T}_t$ $\mathcal{T}_{lla}$ $\mathcal{T}_{rla}$ $\mathcal{T}_n$ $\mathcal{T}_{cxy}$ $\mathcal{T}_{cz}$ $\mathcal{T}_{lf}$ $\mathcal{T}_{rf}$ $\mathcal{T}_{lh}$ $\mathcal{T}_{rh}$ $\mathcal{T}_h$
C2	$\mathcal{T}_t$ $\mathcal{T}_{la}$ $\mathcal{T}_{la}$ $\mathcal{T}_{ll}$ $\mathcal{T}_{rl}$ $\mathcal{T}_n$ $\mathcal{T}_{cxy}$ $\mathcal{T}_{wh}$ $\mathcal{T}_h$ $\mathcal{T}_{lf}$ $\mathcal{T}_{rf}$

3.2.3 Learning Algorithm (offline optimization on simulated robot)

The main features that should characterize our "generic" controller are high-performance (tracking of the reference motions) and robustness (reduction of the tipping moment of the robot during the motion). These two objectives may be antagonistic (e.g., when the desired motions are particularly challenging for the robot balance) hence the problem is naturally posed as a multi-objective optimization problem.

Multi-objective optimization relies on the Pareto *dominance* concept [45]: a solution x_1 dominates x_2 if and only if x_1 is better than x_2 for all the objectives; if x_2 is better for at least one objective, then x_1 and x_2 are equally interesting as they represent different trade-offs. Using this definition, optimizing means finding the set of the non-dominated solutions of the search space, that is, solutions that cannot be improved with respect to one objective without decreasing their score with respect to the other. This set is called the "Pareto front". Although we select a single solution to be used on the robot, the knowledge of multiple Pareto-optimal solutions, "helps the user to compare, choose a trade-off solution, avoiding multiple optimization runs and artificial fix-ups" [45].

The algorithm we opted for is the Non-dominated Sorting Genetic Algorithm II (NSGA II)

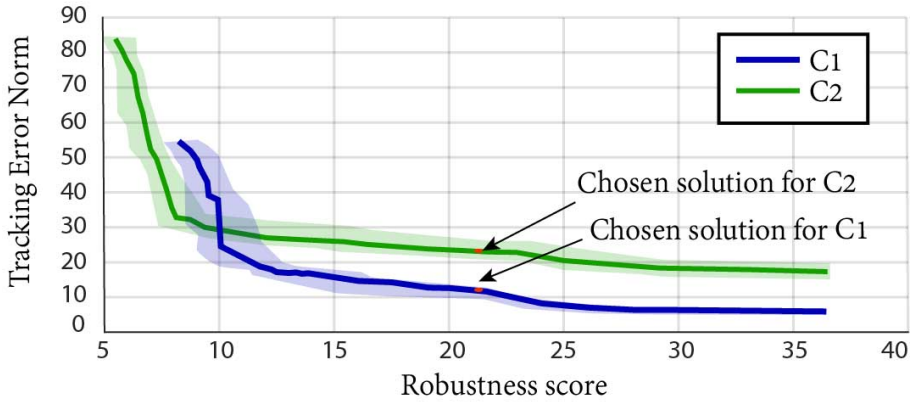


Figure 3.3: Pareto front solution. Median Pareto front (thick line) and associated IQR (colored region) computed by NSGA-II in 20 runs with a population of 100 individuals and 300 generations for the controllers $C1$ (blue) and $C2$ (green).

[44], one of the most efficient stochastic multi-objective optimization methods. The objective functions are the accumulated tracking error f_1, and a robustness score f_2, i.e. a measure of the ZMP position inside the Support Polygon (SP). We additionally considered a third objective function f_3, consisting of a fall avoidance score. f_3 is not strictly necessary since this information is already encoded by f_1 and f_2, but we included it to avoid getting initially stuck in some local optimum, as we explain later on. The goal of the algorithm is to minimize f_1 and f_2 while maximizing f_3

$$\text{Minimize} \quad (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), -f_3(\boldsymbol{x})),$$

where $\boldsymbol{x}$ are the control parameters. These objectives are never combined together, since we use Pareto-based multi-objective optimization. The objective functions are initialized at zero and change at each time step i as follows:

$$\begin{cases} f_{1_i} = f_{1_{i-1}} + (\sum_k \left|\Phi_i^k - \overline{\Phi}_i^k\right|) \\ f_{2_i} = f_{2_{i-1}} + x_{SP} + y_{SP} \quad \text{if (robot fallen)} \\ f_{2_i} = f_{2_{i-1}} + |x_i^{cz}| + |y_i^{cz}| \quad \text{if (robot not fallen)} \\ f_{3_i} = f_{3_{i-1}} - \alpha_3 \quad \text{if (robot fallen)} \end{cases} \tag{3.2}$$

where Φ_i^k and $\overline{\Phi}_i^k$ are respectively the reference and the actual value measured on the simulated robot of task k at time step i; x_i^{cz} is the distance in the frontal direction of the ZMP from the center of the SP at time step i, while y_i^{cz} is the distance in the horizontal direction of the ZMP from the line connecting the feet; x_{SP},y_{SP} are the dimensions of the SP and α_3 an arbitrary positive value. The cost function f_1 includes both position, orientation and postural tracking errors. We combined the Cartesian position errors in cm to postural and orientation errors in degrees, so to consider the tasks of different nature approximately with the same weight in f_1.

When the algorithm generates some parameters that correspond to an unfeasible stack (e.g. empty level in the hierarchy that is not the last) the controller fails, i.e. it cannot be initialized. It can also fail when the solver cannot find a solution for the QP problem without violating its

constraints. In such cases, we add the following penalties:

$$\begin{cases} f_1 = f_{1_{i-1}} + \alpha_{F1}(I - i) \text{ if } (i \neq 0) \\ f_2 = f_{2_{i-1}} + \alpha_{F2}(I - i) \text{ if } (i \neq 0) \\ f_3 = f_{3_{i-1}} - \alpha_{F3}(I - i) \text{ if } (i \neq 0) \\ f_1 = F_{max} \quad \text{if } (i = 0) \\ f_2 = F_{max} \text{ if } (i = 0) \\ f_3 = -F_{max} \quad \text{if } (i = 0) \end{cases} \tag{3.3}$$

where I is the total number of steps in the recorded sequence, i is the time step when the solver fails, F_{max} is the largest value a floating-point variable can hold, α_{F3} can be set to any arbitrary value greater than α_3 and lower than F_{max}/I and α_{F1}, α_{F2} to arbitrary values that are lower than F_{max}/I and respectively greater than the largest increment (Δ_1, Δ_2) that f_1, f_2 could get in a time step when the controller does not fail. Since it might be difficult for the user to determine in advance the lower bound values of α_{F1}, α_{F2}, we consider also f_3 in the optimization to help the algorithm even for a wrong guess by the user of this lower bound of α_{F1}, α_{F2}. If the first generations indeed have large f_1, f_2 scores with α_{F1}, α_{F2} lower than Δ_1, Δ_2, it may happen that the algorithm prefers failing solutions to bad performing solutions that do not fail, hence the controller fails very often and the algorithm struggles to converge to a stable configuration. In such cases, even a solution that ends up in falling represents a temporary dominant solution and can help the algorithm progress toward better solutions. Also, f_3 drives the search toward "not falling" solutions. With the adopted cost functions design, the penalty factors $\alpha_{F1}, \alpha_{F2}, \alpha_{F3}, \alpha_3$ can be set to arbitrary values comprised in between lower and upper bounds that are known, with the exception of the lower bounds of α_{F1}, α_{F2} which can be easily guessed from the problem setting, since the first is related to the maximum tracking error and the second to the maximum robustness score in a single time step[5].

3.2.4 Solution Selection (online tests, on the real robot)

Equations (3.2) and (3.3) show that it is not possible to simultaneously get a bad score for f_3 and a good score on f_1 and f_2, hence all the final dominant solutions composing the Pareto front have $f_3 = 0$, i.e. the robot does not fall. We can then analyze the 2D Pareto front related to f_1 and f_2 (Figure 3.3). We start from the most high-performing solution (i.e. lowest f_1 and highest f_2) and try if the solution works on the real robot. If not, we move progressively to solutions that are less high-performing and more "robust" until we get to a solution that achieves balance on the real robot.

3.3 Results

The experiments were performed with the iCub robot, using 32 DoFs for whole-body control. The whole-body controller was developed using the control software library OpenSoT [73, 155]. We tested our approach on an IK-based implementation of the multi-task QP controller (Section 4.3.1). For the experiments on the real robot, the constraints considered in the QP were joint position and velocity limits, and the center of mass kept inside the support polygon. To learn the control parameters, the robot was simulated using the open-source simulator Dart [104].

[5] If the user chooses values that are too small, the inclusion of f_3 allows the algorithm to converge to a solution anyways, as explained before. At any rate these parameters do not need an iterative tuning.

To generate the training and testing motion sequences for the optimization and validation of the controller, we used our motion retargeting framework (Chapter 2). We recorded the movements of a human equipped with the Xsens MVN motion capture suit, and then we retargeted the motions onto the robot model. For the training set, we recorded a 68-seconds sequence, consisting of a series of whole-body movements that solicit as many body parts as possible, so to generalize well to other motions. For the validation, we recorded three different sequences: $S1$: a squat motion; $S2$: a movement where the robot has to shift completely its weight on the left foot and reach a high position with the left hand; $S3$: a complex movement where the robot has to simultaneously rotate and incline its torso while shifting its weight on the right foot and moving the arms.

To show that our approach does not depend on the number or type of tasks (e.g., Cartesian, postural), we seek solutions for two different types of controllers, namely $C1$ and $C2$. They are specified by the set of tasks considered for their control configuration, reported in TABLE 3.1. $C1$ takes as references (from the motion generator) mainly Cartesian tasks, while $C2$ mostly postural trajectories. Both controllers can be used by the humanoid to realize double support motions, but $C2$ replicates the controller that we have been using for teleoperating the iCub robot to perform human-like motions in the previous chapter, where the postural tasks are critical for imitating the human. For this reason, we will validate both controllers on the real robot with the aforementioned sequences $S1, S2$ and $S3$, but only $C2$ will be further validated for the robot teleoperation.

The parameters of the two controllers $C1$ and $C2$ are learnt by the algorithm NSGA-II implemented in $\text{Sferes}_{\text{v2}}$ [125], a C++ framework for multi-core optimization. We set a population p of 100 individuals with 300 generations g (for a total of 30100 evaluations). To provide statistically significant results, we executed in parallel 20 runs on an Intel®Xeon™ E5-2620 with 32 cores at 2.1 GHz. The parallel optimization takes about 15 hours[6]. The duration of the optimization mainly depends on the length of the learning sequence that has to be simulated for a number of $p \cdot g$ times. We opted for a long learning sequence to find a control configuration that can generalize well to many tasks and that can be found by running the optimization just once. For the penalties in the cost functions we used $\alpha_{F1}, \alpha_{F2}, \alpha_{F3} = 0.5$ and $\alpha_3 = 0.1$ (see Section 3.2.3). The algorithm converges to a set of Pareto-optimal trade-offs among f_1 and f_2 (see Figure 3.3).

Once the Pareto-optimal trade-off solutions are found, we follow the testing procedure of Section 3.2.4 to select the final transferable solutions for $C1$ and $C2$ on the real robot. We use the three test sequences $S1, S2, S3$. We tried on the real robot different solutions from the Pareto front starting from those associated to the lowest tracking error (3.2.4). In the attached video, we show on the learned (median) controller $C1$ with a squat motion with straight torso reference, that these solutions are not robust enough to be transferred onto the real robot, which falls. After trying those with $f_2 = 37$, $f_2 = 32$ and $f_2 = 27$ we found that the median solution with associated robustness score $f_2 = 22$ is transferable to the real robot (see attached video). We report here the structure of the stack of $C1$ that represents the most frequent solution, given $f_2 = 22$ (see TABLE 3.4):

$$\begin{aligned} \overline{\mathcal{S}}_{C1} = (w_f(\mathcal{T}_{lf} + \mathcal{T}_{rf}) + w_h\mathcal{T}_h)/ \\ (w_{cxy}\mathcal{T}_{cxy} + w_{cz}\mathcal{T}_{cz} + w_t\mathcal{T}_t + \\ + w_{ha}(\mathcal{T}_{lh} + \mathcal{T}_{rh}))/ \\ (w_{la}(\mathcal{T}_{lla} + \mathcal{T}_{rla})); \end{aligned} \tag{3.4}$$

while for $C2$ we have:

[6]Note that the optimization is not trajectory-specific and is only run once to get a controller that can achieve many trajectories.

Table 3.2: Feedback Gains (FG) associated to the 20 learned configurations given $f_2 = 22$.

C1			**C2**		
FG	Median	IQR	FG	Median	IQR
λ_{hand}	0.0191	0.033	λ_{waist}	0.5491	0.3791
λ_{feet}	0.0577	0.064	λ_{feet}	0.2486	0.0983
σ_{feet}	0.0051	0.0059	σ_{feet}	0.0983	0.1231
λ_{com}	0.4426	0.1664	λ_{com}	0.5911	0.1946
σ_{head}	0.0796	0.0234	σ_{head}	0.2778	0.2560
$\mu_{posture}$	0.5605	0.2145	$\mu_{posture}$	0.5162	0.0821
			σ_{chest}	0.6052	0.4009

Table 3.3: Soft Priority Weights (SPW) associated to the 20 learned configurations given $f_2 = 22$.

C1			**C2**		
SFW	Median	IQR	SFW	Median	IQR
w_{ha}	0.639	0.2131	w_{ha}	0.8406	0.296
w_f	0.5357	0.1786	w_f	0.5835	0.2144
w_{cxy}	0.8368	0.1941	w_{cxy}	0.9519	0.1984
w_h	0.3343	0.3101	w_h	0.5357	0.2465
w_n	0.3256	0.2893	w_n	0.406	0.2419
w_t	0.9258	0.245	w_t	0.656	0.2138
w_{la}	0.3599	0.3133	w_l	0.1145	0.3756
w_{cz}	0.7684	0.2191	w_{wh}	0.879	0.1785
			w_c	0.9902	0.091

$$\begin{gathered}\overline{\mathcal{S}}_{C2} = (w_f(\mathcal{T}_{lf} + \mathcal{T}_{rf}) + w_n\mathcal{T}_n)/ \\ (w_{cxy}\mathcal{T}_{cxy} + w_{wh}\mathcal{T}_{wh} + w_t\mathcal{T}_t + w_a(\mathcal{T}_{la} + \mathcal{T}_{ra}))/ \\ (w_c\mathcal{T}_c);\end{gathered} \tag{3.5}$$

The corresponding median gains and soft weights are reported in TABLE 3.2 and 3.3, along with the interquartile range (IQR) of these parameters in the 20 learned configurations. TABLE 3.4 indicates the frequency of each task in the different levels of the hierarchy in the 20 runs[7].

We compared the performance of the learned controller for $C1$ with respect to a baseline solution that was manually tuned by an expert. The test was performed on the three different sequences $S1, S2, S3$.

The baseline was the following Hand-Tuned controller (HT):

$$\begin{gathered}\mathcal{S}_{HT} = (w_{ht}(\mathcal{T}_{lf} + \mathcal{T}_{rf}) + \mathcal{T}_{cxy} + \mathcal{T}_h)/ \\ (\mathcal{T}_{cz} + \mathcal{T}_t + (\mathcal{T}_{lh} + \mathcal{T}_{rh}))/ \\ ((\mathcal{T}_{lla} + \mathcal{T}_{rla}));\end{gathered} \tag{3.6}$$

where the the tasks concerning the feet contacts $\mathcal{T}_{lf}, \mathcal{T}_{rf}$ and the center of mass $\mathcal{T}_{cxy}$ are reasonably considered at the highest level of priority, being the most relevant for balance.The HT solution was validated on the robot for double support motions. Soft priorities were all set to 1. The FG

[7]The purpose of the 20 runs is to provide statistical evidence about the convergence of the multi-objective optimization algorithm. We can notice that the variability of the Pareto front is small (see IQR region in Figure 3.3), meaning that for a real use-case only one run is sufficient to compute the Pareto front and to find the Pareto-optimal solutions.

Table 3.4: Task frequency in each level of the hierarchy in the 20 learned configurations, given $f_2 = 22$. The most frequent solution is in bold.

C1				**C2**			
Task	$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_3$	Task	$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_3$
$\mathcal{T}_{lh}$, $\mathcal{T}_{rh}$	2	**18**	0	$\mathcal{T}_{la}$, $\mathcal{T}_{ra}$	3	**17**	0
$\mathcal{T}_{lf}$, $\mathcal{T}_{rf}$	**18**	2	0	$\mathcal{T}_{lf}$, $\mathcal{T}_{rf}$	**19**	1	0
$\mathcal{T}_{cxy}$	4	**16**	0	$\mathcal{T}_{cxy}$	3	**17**	0
$\mathcal{T}_h$	**11**	0	1	$\mathcal{T}_h$	6	1	0
$\mathcal{T}_n$	9	1	1	$\mathcal{T}_n$	**9**	4	0
$\mathcal{T}_t$	3	**17**	0	$\mathcal{T}_t$	1	**17**	2
$\mathcal{T}_{lla}$, $\mathcal{T}_{rla}$	1	3	**16**	$\mathcal{T}_{ll}$, $\mathcal{T}_{rl}$	0	0	5
$\mathcal{T}_{cz}$	4	**16**	0	$\mathcal{T}_{wh}$	3	**17**	0
				$\mathcal{T}_c$	0	1	**9**

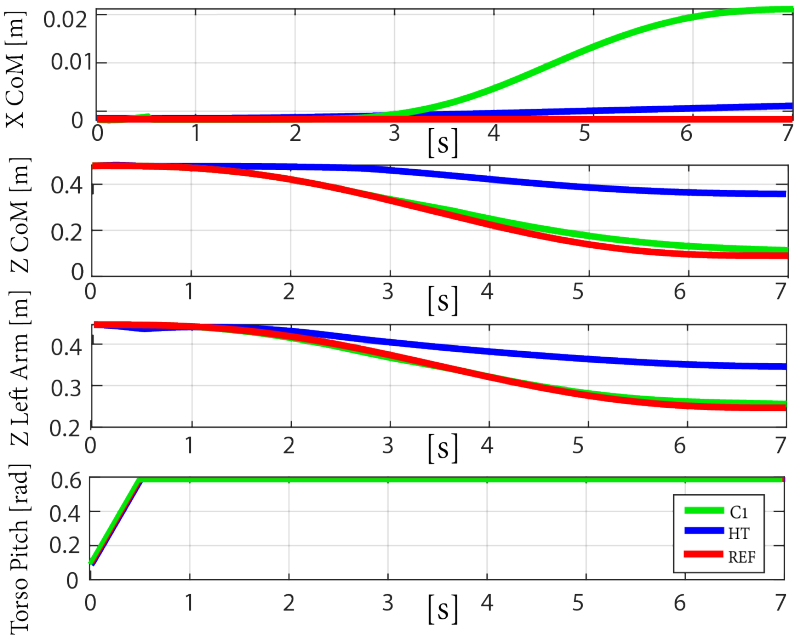


Figure 3.4: Comparison of the tracking performance of the median learned controller $C1$ and the hand-tuned controller HT (with $w_{ht} = 1$) in sequence $S1$, on some significant tasks (in simulation).

were all set to 0.1 except for λ_{feet}, σ_{feet}, λ_{com} that were set to 1 as done in previous work where our optimization approach was not yet available [73, 155, 181].

Figure 3.4 and the video (youtu.be/RJW67SU6Yf0, 6min47s-7min29s) show respectively in simulation and on the real robot, how the median learned controller $C1$ outperforms HT in tracking the sequence $S1$. Since simultaneous mastering of multiple tasks during motion can be challenging (especially if these references are computed on approximate models that do not match reality), some tasks can be tracked with less precision than others. Indeed, the optimization algorithm found that penalizing the tracking of a not optimal reference for the x position of the CoM,

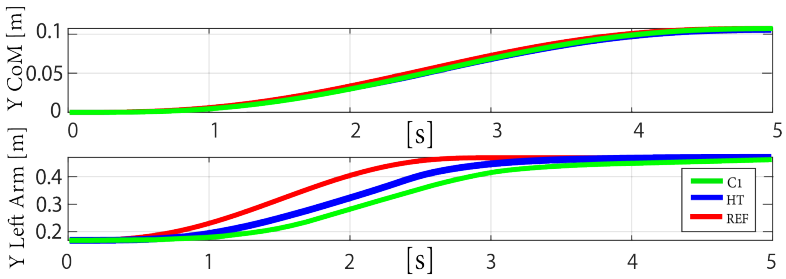


Figure 3.5: Comparison of the tracking performance of the median learned controller $C1$ and the hand-tuned controller HT (with $w_{ht} = 0.7$, since the controller cannot find a solution for $w_{ht} = 1$) in sequence $S2$, on some significant tasks (in simulation).

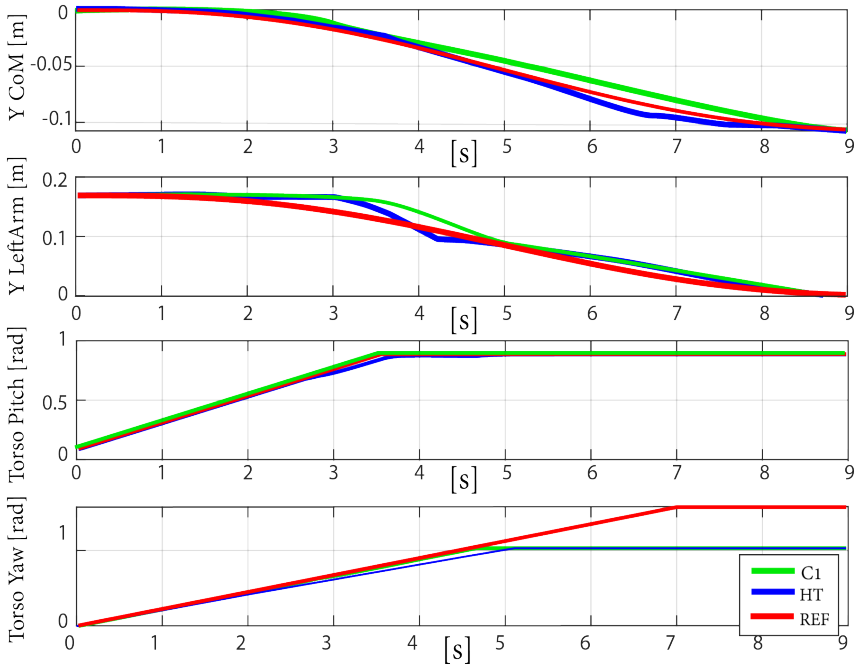


Figure 3.6: Comparison of the tracking performance of the median learned controller $C1$ and the hand-tuned controller HT (with the same task priorities of $C1$) in sequence $S3$, on some significant tasks (in simulation).

can then lead to a better overall whole-body tracking performance, especially a better tracking of the height of the CoM (see Figure 3.4). In $S2$, by setting $w_{ht} = 0.7$, we could get with HT, a performance that is comparable to the median learned controller $C1$ in simulation (see Figure 3.5). However, when transferring the result onto the real robot, HT made the robot fall (due to a not robust enough choice of the priorities and FG) while $C1$ did not, as shown in the attached video. In $S3$, HT failed to find a solution, for both $w_{ht} = 0.7, 1$. We then considered the same structure of the stack and the same soft weights of $C1$ in HT to see how only the FG can affect the tracking performance (Figure 3.6). Also in this case, with the gains set manually (that are not "robust") the real robot fell, while with the optimized gains it kept the balance (attached video).

To show how a controller optimized through our approach can be used for generic motions – not previously encountered in the learning sequence, though from the same category (i.e., double

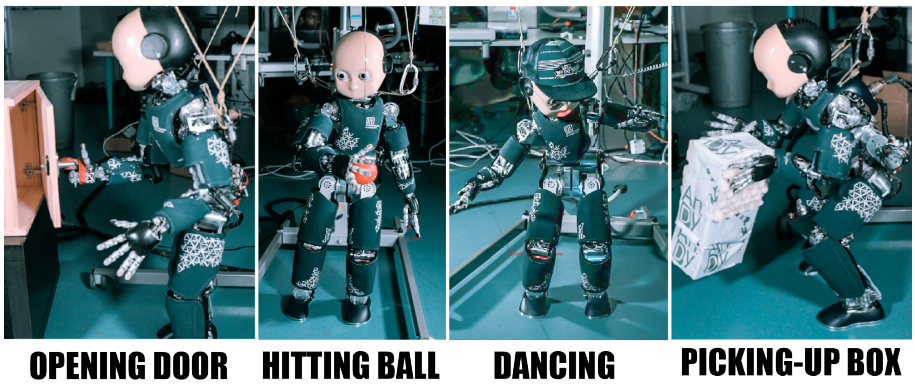


Figure 3.7: The iCub robot controlled with the learned configuration $C2$ while performing different teleoperated tasks.

support) – we used the optimized controller of $C2$ in our teleoperation system. We teleoperated the iCub to perform several actions: picking up a box, pushing a ball in a box, pushing aside a box, opening and closing the door of a container, dancing and hitting a ball (see Figure 3.7). The human operator generated the movements in real-time, that were tracked by the Xsens MVN suit and retargeted to the robot. The teleoperated sequence, lasting more than 2 minutes of continuous movements, is reported in the video youtu.be/RJW67SU6Yf0 at 7min53s.

3.4 Discussion

We proposed a framework to automatically learn both the structure and the parameters of a "generic" whole-body controller. We used multi-objective optimization to look for solutions on the Pareto front, representing optimal trade-offs of performance (tracking the reference trajectories) and robustness (limiting the tipping moment of the robot). This allowed us to efficiently find a set of Pareto-optimal trade-off solutions in simulation. The user can conveniently choose from the Pareto-front a pre-optimized solution to test on the real robot: this reduces considerably the number of test trials on the real robot and improves the parameters tuning. To give the reader an example, to find a suitable controller configuration for humanoid teleoperation it only took us three test trials on the real robot. Before our method, tuning such a controller was a time consuming trial-and-error procedure.

In this work we limited our approach to double support motions with a simple QP controller structure that does not include friction cones, contact wrenches and centroidal dynamics in the constraints [103]. Those will be added in future work, to deal with more complex and dynamic motions. In this case other robustness criteria would have to be investigated, since a conservative ZMP distance from the center of the support polygon would not be sufficient to guarantee the stability of the contacts. A possibility could be to use a robustness score that maximizes the distance of the ground reaction forces from and within the friction cone bounds. Also recurring to some domain randomization in the simulation roll-outs, especially involving the friction at the level of the contacts and of the robot actuators, could help obtain more robust solutions.

4

Multi-mode Teleoperation System

We envision a world where humanoid robots can be used as human avatars to replace humans in dangerous situations that can put their lives at serious risk. The intuition and intelligence of human operators can be leveraged to make humanoids perform complex tasks, provided that suitable control interfaces and teleoperation modes are designed. In Chapter 2, we showed how the similar human-morphology of humanoids can be exploited to intuitively control their whole-body. However, in locomotion tasks like walking, the differences in the body structure and dynamics between human and robot become too significant to be addressed in the same manner. Hence in this chapter, we present a teleoperation framework for executing loco-manipulation tasks with a humanoid that provides two different modes of teleoperation:

– a high-level teleoperation setting in which the operator uses a joystick to send reference commands to the robot such as direction and velocity of motion, without dealing with its actual execution;

– a low-level teleoperation setting in which the operator generates whole-body movements for the robot by means of a motion capture suit (Chapter 2).

In both cases the human operator receives visual feedback through a virtual reality (VR) headset connected to the cameras of the robot avatar.

The work presented in this chapter was published in [184]. It was the result of a collaboration with the colleagues from University La Sapienza of Rome, who implemented the MPC-based gait generator [165]. Our personal contribution consisted in embedding the MPC into the teleoperation system, providing a unique framework that allows to switch between different teleoperation modes.

In the following, we first review the state of the art of teleoperation systems supporting locomotion, then we present our framework and we demonstrate it on the humanoid robot iCub and on the simulated robot HRP-4C.

4.1 Related work

The idea of teleoperating robots with VR was first proposed by Tachi [176]. The retargeting of the upper-body joints, which is important for manipulation, has often been done independently from the motion generation of the legs, crucial for balancing and locomotion. In [109] for example, the authors employ the mobile manipulator Justin to retarget upper-body motions with haptic feedback at the hands, without considering leg motions (Figure 4.1).

Kim *et al.* [97] were among the first to extend the robot teleoperation to walking motions. In [97], upper-body motions and walking are separately retargeted onto the humanoid robot MAHRU by using a wearable motion capture system. Whenever the operator walks, this triggers a human-independent walking by the robot. For the retargeting of upper-body motions only the arms are involved. Instead, in [77], Hu *et al.* focus on teleoperating exclusively the walking for the humanoid TORO, but considering also the human footsteps and configuration of the leg joints in the retargeting.

Figure 4.1: Examples of teleoperated locomotion. Top-left: iCub robot teleoperated with a VR walking platform [55]; top-right: walking imitation of MAHRU robot [97]; bottom-left: whole-body teleoperation of a stepping motion with the robot JAXON [79]; bottom-right: remote guidance of the walking of the Valkyrie robot through a GUI [87].

In [55], the authors teleoperate the iCub robot in an immersive scenario using a VR headset and a walking platform (Figure 4.1). The robot starts and stops walking whenever the operator does, but the retargetable double-support motions are only limited arm movements.

Motion retargeting can be performed also at whole-body level. Ishiguro *et al* [79] conducted some experiments retargeting highly dynamic upper-body and leg motions onto the humanoid robot JAXON (Fig 4.1). Although suitable for executing challenging movements, such as kicking or hitting a tennis ball with a racket, their technique cannot be used for extended walking due to the great mismatch between the human and robot dynamics, which forces the human to walk in a

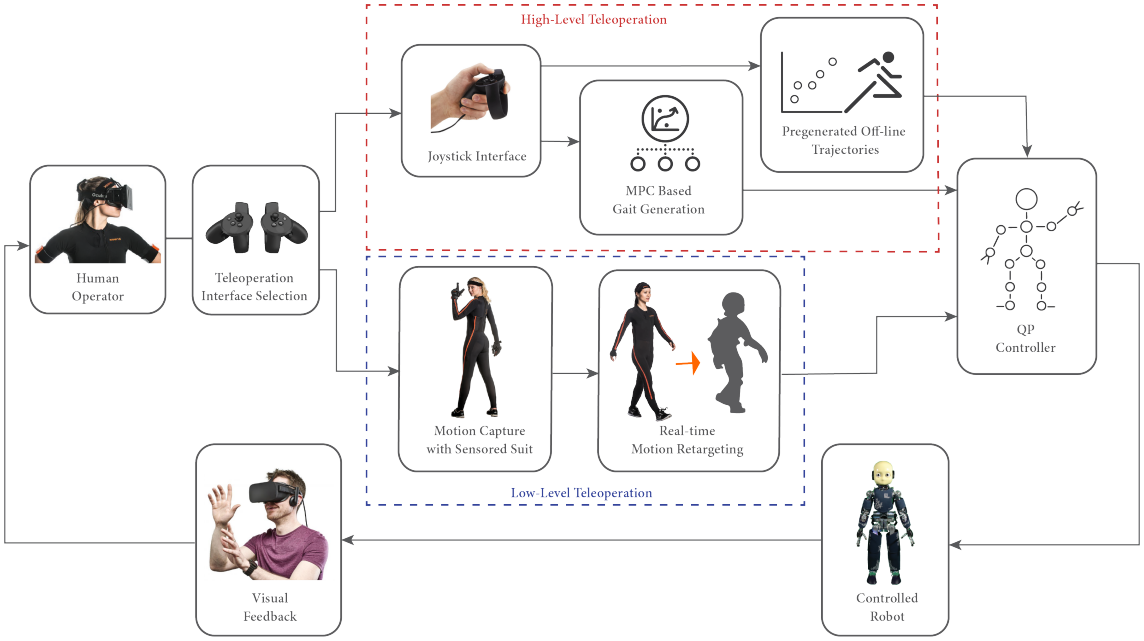


Figure 4.2: Overview of the proposed teleoperation system. The user can choose between two operational modes, i.e., high- or low-level teleoperation.

very unnatural way.

The main challenge of teleoperating highly dynamical motions is to ensure smooth and stable motions in real-time while guaranteeing the robot's balance. Similar issues must be addressed in robotic walking, where a widespread approach to generate robust dynamic motions exploits Model Predictive Control (MPC) on reduced models of the robotic system. For gait generation the most common strategy relies on the concept of Zero Moment Point (ZMP), i.e., the point with respect to which the horizontal momenta of the ground reaction forces are zero. Dynamic equilibrium is guaranteed by keeping the ZMP at all times within the robot support polygon, i.e., the convex hull of the contact points.

Many successful techniques for generating stable gaits are based on a simplified linear dynamic model [90] relating the ZMP to the center of mass, derived by neglecting any rotational contribution around the center of mass which is also assumed to be at constant height. This model is called the Linear Inverted Pendulum (LIP) or the Cart-Table (CT), depending on whether the ZMP is treated as an input or an output.

In [88], the Cart-Table model is used to design a linear quadratic controller with preview. Constraints were added in [194] leading to an MPC formulation and also allowing the automatic choice of the footsteps [71]. To cope with the unstable nature of the LIP, an explicit stability constraint ensuring that the center of mass trajectory is bounded w.r.t. the ZMP has been introduced in the MPC design in [164]. Extensions to walk-to locomotion [18] or uneven ground [197] have also been proposed.

Another approach that is often employed in an MPC formulation is the Divergent Component of Motion (DCM) [179], which can be viewed as an extension of the capture point concept to the three-dimensional case for uneven terrain [56]. The DCM represents the point where the center of mass of the of the unstable LIPM converges. Therefore, the main goal of this approach is to implement a control law that stabilizes the unstable LIPM dynamics, as well as the ZMP. This approach is used in [55] as an MPC formulation to compute the footsteps and center of mass

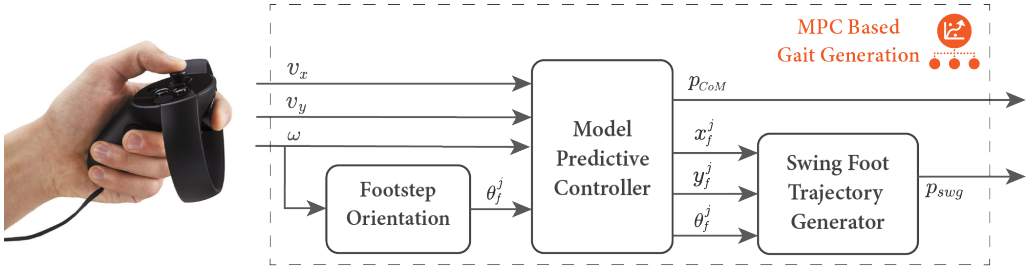


Figure 4.3: Pipeline of the MPC-based gait generation module.

trajectory of the robot, which are then fed to an IK QP solver.

Differently from previous work, our teleoperation framework allows us to alternate walking motions that are generated by an LIPM-based MPC gait generator, to whole-body manipulation, not limited to constrained arm movements.

4.2 Overview of the System

Our framework is illustrated in Figure 4.2. The human operator can choose between two different teleoperation modes using the buttons of the VR controller: a *low-level* mode for full real-time control of the robot via motion retargeting and a *high-level* mode to command walking or pre-optimized task trajectories. Both modalities share the same whole-body controller for computing in real-time the commands to be sent to the robot.

In the first operational mode, the human posture is tracked by a motion tracking system, in our case the Xsens motion capture suit. The data are then mapped to feasible corresponding joint values for the robot. To achieve dynamical balance, the references are corrected by a stabilizer and then fed to the whole-body robot controller. The latter is formalized as a multi-task QP controller, where the task references are the desired robot posture and its stabilized center of mass.

The second operational mode is characterized by a higher degree of autonomy of the robot, and it is triggered by the operator using the joystick. For walking, the operator gives a direction and velocity references for the humanoid gait through the analog sticks. These are translated into a timed sequence of footsteps, swinging foot and center of mass trajectories through an MPC-based control scheme. Alternatively, the operator uses the joystick buttons to select the one among different pre-defined task trajectories. The corresponding trajectories, which have been pre-optimized off-line [68], are sent as reference to the robot controller and then simply reproduced.

The reason for the two distinct modes is that retargeting is essentially kinematic and it is not effective for on-line teleoperation of dynamic motions such as walking or stepping. In our experience, retargeting of walking is not a viable solution for many reasons: first, the stride of the robot is typically shorter than the operator's; second, the robot foot trajectory is often optimized for balance and impacts, while the trajectory retargeted from the human is not compliant with these requirements; third, since humanoids cannot walk as fast as humans, in a retargeting context the operator would be forced to walk in an unnatural way, ultimately leading to inefficient robot locomotion. For these reasons, it is better to rely on a MPC-based gait generation and avoid retargeting altogether in this phase. Similar considerations can be made for many motion primitives or task trajectories that impact or leverage the robot dynamics, such as stepping or serving in tennis: in our view, this is the kind of motions that should be pre-optimized off-line as they are specific to the robot dynamics.

4.3 Methods

4.3.1 Whole-body controller

The whole-body motion of the robot is defined by the following trajectories: center of mass ground projection (com_g), waist height ($base_h$), hands positions ($lhand$ and $rhand$), arms postures ($larm_q$), neck posture ($neck_q$), torso posture($torso_q$) and feet poses ($lfoot$ and $rfoot$). Given these reference trajectories, the robot commands $\boldsymbol{q}$ are generated by solving the redundant inverse kinematics, which can be formulated as a constrained quadratic programming problem with equality and inequality constraints as explained in Section 1.5.1.

The stack of tasks we used is different than that from the experiments presented in Section 2.3, since we had to take into account also the walking of the robot. In this case the center of mass position and the feet poses have to be given the highest priority:

$$stack = (lfoot + rfoot + com_g)/$$
$$(0.5base_h + neck_q + 0.5torso_q + 0.1larm_q + 0.1rarm_q + lhand + rhand);$$

The priorities are not optimal and were manually tuned without using our multi-objective stochastic optimization presented in Chapter 3, because we had not implemented it yet at the time. The inequality constraints of the QP problem, contain the robot joint velocity bounds and zero moment point bounds, which is constrained to stay inside the support polygon.

The controller is based on the OpenSOT framework [155] and is run at 100 Hz.

4.3.2 Low-level teleoperation mode

The motion retargeting in the low-level teleoperation mode is built on our previous work presented in Chapter 2. For convenience, we summarize here the methodology previously introduced. Joint positions are measured and grouped into subcategories: head, torso, left arm, right arm, left leg and right leg. Also the ground projection of the center of mass, the height of the waist, the orientation of the head and the position of the feet are controlled.

In the joint retargeting module, the Xsens skeleton degrees of freedom are assigned to the corresponding ones of the iCub robot. Then, we consider the joint angle variations of the human with respect to the starting posture to compute the corresponding instantaneous values of the robot joint angles.

The same approach is used to retarget the relative Cartesian position of a body segment with respect to a base link, with the difference that the variation of the human positions has to be properly scaled by the human-robot limb length ratio..

To track the human center of mass we use normalized offsets, from which we then reconstruct the robot center of mass ground position.

The resultant retargeted motion it is not guaranteed to be dynamically balanced and different methods can be used to correct it. In our teleoperation approach we want a dynamically balanced center of mass trajectory and we adopt the LIP model to properly modify the reference trajectory.

The stabilized center of mass reference, Cartesian tasks and postural tasks are set as reference tasks in the multi-task QP controller.

4.3.3 High-level Teleoperation Mode

As illustrated in Figure 4.2, the humanoid desired motion is either defined offline (pre-generated task trajectories or motion primitives [68]) or online (MPC-based gait generator).

Pre-generated trajectories are triggered by the operator depending on the situation. For example, in the final phase of the reported experiment the robot moves its feet apart autonomously, independently of the human's lower limbs motion, for picking up the box more effectively.

Other tasks or repetitive movements like standing up, can be recorded off-line and replicated for a quick and precise execution. We also recommend the use of this option to make the robot perform motions that are not ergonomic for the operator or can be uncomfortable to perform. Note also that the execution of pre-defined motions may be more convenient in the presence of signal degradation or delay.

The online gait generation is based on the MPC scheme proposed in [164, 165] and is summarized in Figure 4.3. The joystick provides reference velocities (v_x, v_y) and ω for respectively the sagittal, coronal and angular motions. Footstep orientations are computed in a separate stage and used as known parameters in the next module. This is useful to guarantee linearity of the constraints in the following MPC formulation.

To generate the footstep orientations a first QP problem

$$\text{(QP1):} \qquad \min_{\Theta_f^k} \sum_{j=1}^{F} (\theta_f^j - \theta_f^{j-1} - \omega T_s)^2$$

$$\text{subject to} \quad |\theta_f^j - \theta_f^{j-1}| \leq \theta_{\max}$$

is solved for all the F predicted footsteps that fall within the prediction horizon of the MPC problem. These are denoted as $\Theta_f^k = (\theta_f^1, \ldots, \theta_f^F)$, while T_s is the step duration and $\theta_{\max}$ the maximum allowed rotation between two consecutive steps.

A second module based on MPC generates the center of mass trajectory and the positions of the footsteps, while their orientations are inherited from the first module. At each time instant t_k, the MPC solves a QP problem on a prediction horizon $[t_k, t_{k+C}]$ based on a prediction model and constraints. The sampling interval has duration δ. We use as prediction model the LIP (2.14) with an additional integrator on the input (dynamic extension), so that the ZMP velocities $(\dot{x}_{ZMP}, \dot{y}_{ZMP})$ are used as control input. The decision variables of the QP problem are therefore $\dot{x}_{ZMP}^k, \ldots \dot{x}_{ZMP}^{k+C-1}$, $\dot{y}_{ZMP}^k, \ldots \dot{y}_{ZMP}^{k+C-1}$ and the footstep positions $\dot{x}_f^1, \ldots \dot{x}_f^F, \dot{y}_f^1, \ldots \dot{y}_f^F$, which are collected in the vector U^k of decision variables. The ZMP velocities are assumed to be constant in each sampling interval $[t_i, t_{i+1}]$ resulting in a piece-wise linear ZMP.

The constraints enforced in the second QP are:

– a *balance* constraint, which ensures that the ZMP is at all times inside the support polygon;

– a *kinematic* constraint, guaranteeing that the footsteps are placed in a region which is kinematically feasible, and avoid self-collisions;

– a *stability* constraint, which makes sure that the generated CoM trajectory does not diverge with respect to the ZMP.

The balance or ZMP constraint at a generic instant of the single support is expressed as

$$R_j^T \begin{pmatrix} x_{ZMP}^{k+i} - x_f^j \\ y_{ZMP}^{k+i} - y_f^j \end{pmatrix} \leq \frac{1}{2} \begin{pmatrix} d_{z,x} \\ d_{z,y} \end{pmatrix} \tag{4.1}$$

where $d_{z,x}$ and $d_{z,y}$ denote the size of the rectangular support polygon while $(x_{ZMP}^{k+i}, y_{ZMP}^{k+i})$ is the ZMP position at the i-th prediction instant. R_j^T is the rotation matrix associated with the orientation of the j-th predicted footstep within the prediction horizon, which is computed in the

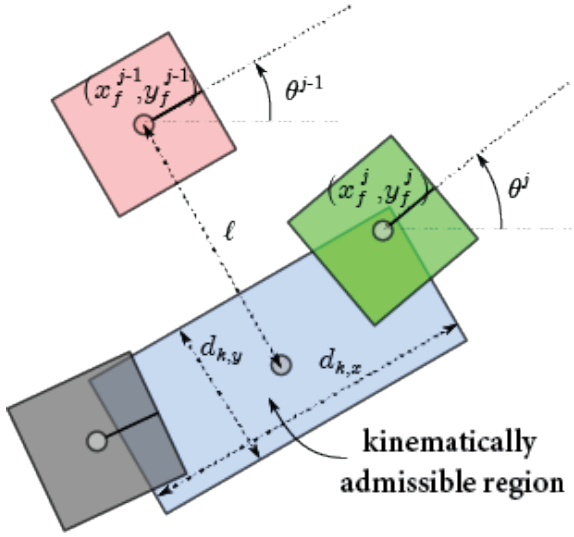


Figure 4.4: Illustration of the kinematic constraint. The center of the green footstep can be placed in the blue region, which is defined with respect to the previous footstep (red).

Footstep Orientation block of Figure 4.3. The ZMP constraint is enforced at each instant of the prediction horizon, i.e., for $i = 1, \dots, C$.

The kinematic constraint ensures that the footsteps are placed consistently with the robot capabilities. In particular this involves avoiding footsteps which are too far away and thus unreachable, or too close which would cause self collision between the two legs. The constraint is

$$R_{j-1}^T \begin{pmatrix} x_f^j - x_f^{j-1} \\ y_f^j - y_f^{j-1} \end{pmatrix} \leq \pm \begin{pmatrix} 0 \\ \ell \end{pmatrix} + \frac{1}{2} \begin{pmatrix} d_{k,x} \\ d_{k,y} \end{pmatrix} \tag{4.2}$$

where $d_{k,x}$ and $d_{k,y}$ represent the size of a rectangular region which is kinematically feasible and avoids self-collisions, and ℓ the position of its center w.r.t. the previous footstep. The $\pm$ sign regularly alternates discriminating between left and right footsteps. Figure 4.4 shows a visual representation of the kinematic constraint.

To understand the necessity of the stability constraint, note that the LIP model (2.14), and hence the prediction model, has an unstable eigenvalue; therefore, the generic center of mass trajectory will in general diverge w.r.t. the ZMP. There exists however a *stability condition*, relating the center of mass initial state in t_k to the future ZMP, expressed as

$$x_{CoM}^k + \frac{\dot{x}_{CoM}^k}{\eta} = \eta \int_{t_k}^{\infty} e^{-\eta(\tau - t_k)} x_{ZMP}(\tau) d\tau \tag{4.3}$$

which ensures that the center of mass trajectory remains bounded with respect to the ZMP. A similar expression holds along y.

The stability constraint of the MPC is obtained by computing (4.3) for a piece-wise linear ZMP trajectory. However, note that the integral requires the future ZMP trajectory, which is available only up to the prediction horizon. The remaining part, after t_{k+C}, can be conjectured by using the available information (e.g., planned reference velocities) beyond the prediction horizon. One possibility is to use an *infinite replication* of the control inputs over the horizon, which is especially appropriate for regular gaits that exhibit a periodic behavior; the resulting stability constraint is

$$\sum_{i=0}^{C-1} e^{-i\eta\delta} \dot{x}_{ZMP}^{k+i} = \eta \frac{1 - e^{-C\eta\delta}}{1 - e^{-\eta\delta}} (x_{CoM}^k + \frac{\dot{x}_{CoM}^k}{\eta} - x_{ZMP}^k) \tag{4.4}$$

The second QP problem can finally be stated as

$$\text{(QP2):} \quad \min_{U^k} \sum_{i=0}^{C-1} \Big(\left(\dot{x}^k_{ZMP}\right)^2 + \left(\dot{y}^k_{ZMP}\right)^2 + \beta_x \left(\dot{x}_{CoM} - v_x \cos(\omega\delta i) + v_y \sin(\omega\delta i)\right)^2 + \beta_y \left(\dot{y}_{CoM} - v_x \sin(\omega\delta i) - v_y \cos(\omega\delta i)\right)^2 \Big)$$

subject to:

- ZMP constraints (4.1)
- kinematic constraints (4.2)
- stability constraint (4.4) for x and y

In the cost function, β_x and β_y represent the weights associated with the velocity tracking terms.

Once both QP problems are solved, the first value of the ZMP derivative $(\dot{x}^k_{ZMP}, \dot{y}^k_{ZMP})$ is used to compute the center of mass trajectory $\mathbf{p}_{CoM}$ through the prediction model, while the first predicted footstep $(x^1_f, y^1_f, \theta^1_f)$ is used to generate, using a predefined polynomial shape, a swing foot trajectory $\mathbf{p}_{swg}$ ending at the predicted footstep. The swing foot trajectory is generated using a using a predefined polynomial shape. Both $\mathbf{p}_{CoM}$ and $\mathbf{p}_{swg}$ are finally tracked by the kinematic controller.

4.4 Experiments

We report on an illustrative experiment performed with a human operator and the iCub humanoid robot[8]. In our presented scenario shown in Figure 4.5, the robot must walk and pick a box on the floor to hand it to the worker. The operator is equipped with a wearable motion capture Xsens MVN suit and the VR Oculus Rift system composed by a headset and a pair of joysticks. The suit provides real-time estimation of the posture, the headset gives visual feedback from the robot cameras, while the joysticks allow the operator to switch among the two different teleoperation modes and to guide the robot.

As a first step, the robot should autonomously walk to the box, guided by the operator. After selecting the high-level mode on the VR controller, the robot walks receiving velocity references commands through the joystick. Figure 4.6 shows the MPC-generated CoM trajectory (to be sent to the whole-body controller) together with the footsteps.

When the robot gets in front of the box, the operator stops it. Then, still in high-level mode, the operator prepares himself for the pickup by moving his feet apart, and selects a pre-defined motion for the humanoid to do the same movement independently.

Finally, the operator switches to the low-level mode by using the VR controller and performs a squat motion to pick up the box. Motion retargeting makes the robot follow in real time the movements of the operator, successfully handing the box over to the human. Balance is maintained throughout this phase thanks to the stabilization performed on the retargeted human references, as shown in Figure 4.7. A video clip of this experiment is available online at youtu.be/aCXoDCMd_v4.

[8] iCub is only 104 cm high and cannot lift heavy weights, but our methods are not iCub-specific and could be easily applied to adult-sized humanoids with heavier payloads.

Figure 4.5: Snapshots from the teleoperation experiment with the iCub robot. Top: robot walking in the high-level mode. Bottom: robot controlled in the low-level mode with a motion capture suit and a VR headset.

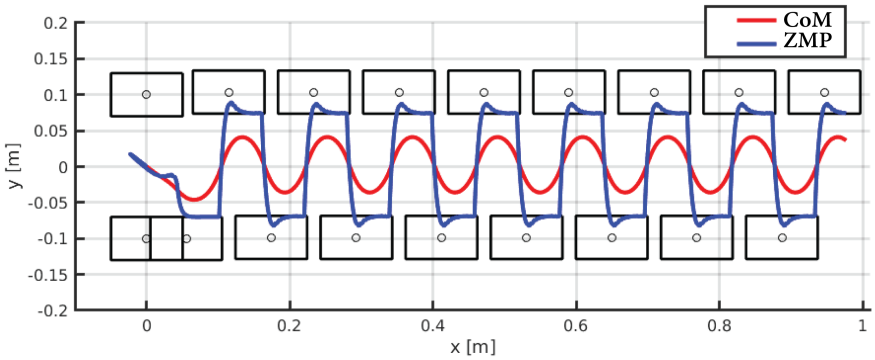


Figure 4.6: Walking phase: CoM and ZMP trajectories along the MPC generated gait.

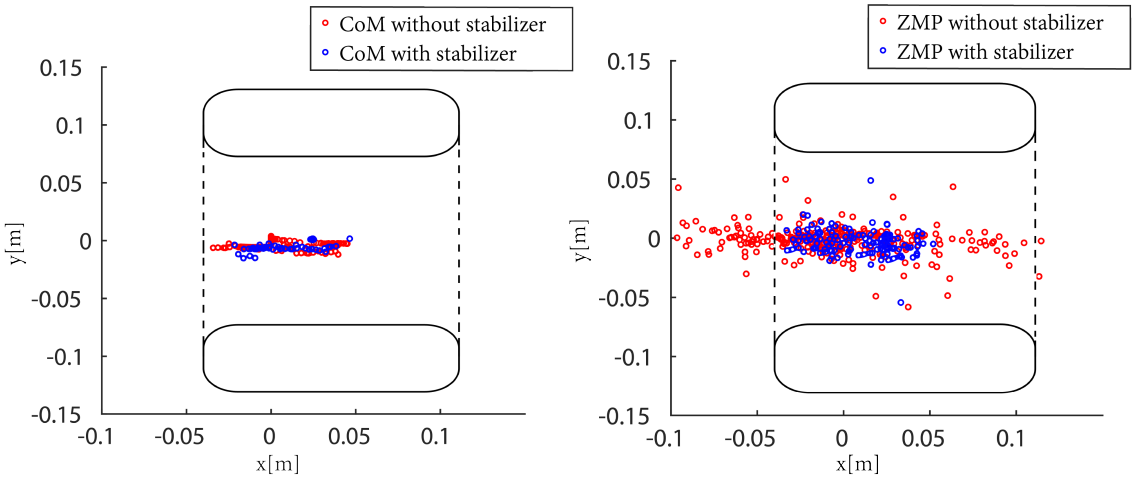


Figure 4.7: Pickup phase: CoM and ZMP trajectories of the robot in double support. The feet of the robot are shown in black. Top: CoM reference position reconstructed from the human reference (red) and corresponding stabilized value (blue). Bottom: ZMP of the LIP model associated to the CoM reference without (red) and with (blue) stabilization.

Experiments with the HRP-4C robot

We implemented the same multi-mode system on the HRP-4C robot by using the whole-body admittance controller [33] implemented within the mc_rtc framework [8] and the MPC-based gait generator provided within the framework [195]. The whole-body admittance controller implements feedback control of the desired force targets issued by DCM feedback and wrench distribution (see Section 1.5.1), while otherwise following the position targets prescribed by the walking pattern given by the MPC [195]. The wrench distribution among contacts is formulated as a QP with ZMP constraints and friction cone constraints. The computed wrenches are fed into a whole-body admittance controller that computes both end-effector and center of mass trajectories. Then based on these, the controller computes the robot joint velocities and accelerations by solving a IK-based QP problem (Section 1.5.1). More details about the controller can be found in [33]. The considered tasks are the poses of the hands, the poses of the feet, the center of mass position, and the whole posture of the robot. The tasks are considered in a weighted combination, where the center of mass and the feet are given weight $w_{CoM,feet} = 1$ since have the highest priority, while the hands' poses have relative weight $w_{hands} = 0.5$ and the postural task has $w_{posture} = 0.05$. These weights have been approximately hand-tuned by an expert user and have not been validated on the real robot.

The system has been tested in simulation alternating walking motions guided trough a joystick interface and whole-body motions pre-recorded off-line, captured with the Xsens system (Figure 4.8). The video is available at mybox.inria.fr/f/bce3ac69c4954c4d95ed/?dl=1.

4.5 Discussion

We proposed a multi-mode teleoperation framework for a humanoid robot on loco-manipulation tasks. The first mode is a low-level teleoperation of all the joints of the robot, while the other enables the execution of high-level commands and pre-designed motion primitives, which can be useful for locomotion or other specific tasks.

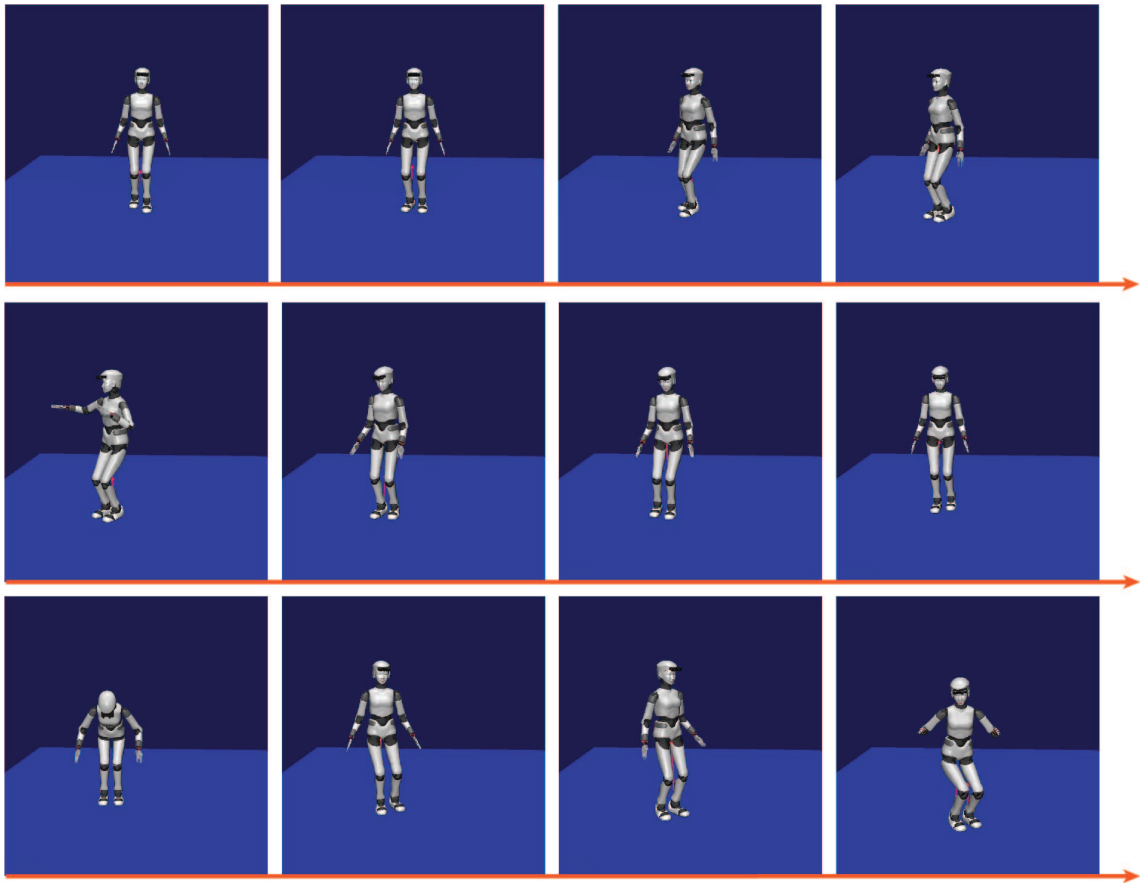


Figure 4.8: Snapshots from the teleoperation experiment with the simulated HRP-4C robot.

Our system is rather intuitive for the user, although proper user studies should be conducted to assess the ease of use of the system and it should be compared to other alternative approaches, including the use of a walking platforms [55], or of full exoskeleton cockpits [80].

The use case presented in the experiment consisted in walking to a box on the ground using the high-level mode, and then picking it up by switching to the low-level mode. It is possible to envision several scenarios in which the presented framework might be employed. For example, the operator might switch to the low-level mode to use control panels, open doors or recover items, all actions that could be necessary during exploration. The robot might also be equipped to execute specific actions necessary for maintenance operations (e.g., tightening screws, assembling parts, etc...) using specialized tools activated from the joystick. Furthermore, the high-level mode of our system relies on a MPC formulation that is valid only for walking on a flat ground. In the case of walking on uneven terrains and 3D locomotion, extensions of the proposed formulation have to be taken into consideration [18, 197].

A future integration would be to employ a microphone at the user side. This could be used to provide vocal commands to the robot, making it perform certain tasks autonomously, or to simply connect the user's voice to the robot's microphone in a telexistence context where the user can communicate through the robotic avatar with other people at the robot side.

5

Delay Compensation in Teleoperation Systems

In a teleoperation scenario, the user and the humanoid robot naturally exchange information through a *communication channel* [29]. However, the communication channel could be non-ideal, introducing transmission delays that impact the stability and performance of the teleoperation. This is the case of many real-world applications such as space exploration, disaster-response scenario, or offshore surveillance. Addressing the delays in space teleoperation is certain the most challenging. In this chapter we propose a novel solution, based on the operator's motion prediction, for coping with the unavoidable communication delays between the movement of the operator and the feedback from the robot. We first review the existing methods for delay compensation and then present our system together with the results obtained from the experiments with the iCub robot.

5.1 Related Work and Contributions

The first studies with robot manipulators [63] suggested that the typical user behavior in a teleoperation system with time delays is to adopt a "move-and-wait" strategy to avoid overcompensations for the delayed perceived errors. However, follow-up experiments [64] showed that this strategy is not effective, even with time delays around 0.3s. This is in sharp contrast to the typical delays that are considered in real experiments aiming at space teleoperation and in robotics competitions, for instance 1 second on average (30s maximum) during the DARPA Robotics Challenge [26], 10 seconds (20s maximum) for the NASA Space Challenge [9], or 0.8s (3s maximum) in the METERON project [107], during which a robot was teleoperated on Earth from the International Space Station.

Time-delayed teleoperation almost exclusively uses *predictive displays* [72, 142], which are virtual displays representing a model of the robot and its environment, to compensate for the delayed visual feedback. In these systems, the operator performs the task on the display, controlling the simulated robot without any time delay, while sending the same commands – which will be delayed – to the real robot. More complex predictive displays overlay the predicted graphics on the delayed video [28, 122], which can be achieved by identifying a model of the environment and transmitting it back to the operator side [122]. However, in most cases, the predicted graphics and the delayed video signals are kept in different displays, due to the difficulty of mixing video and graphics with enough quality and robustness. In all these cases, since perfect modeling is impossible, there is always going to be a difference between the real and modeled environment, which has

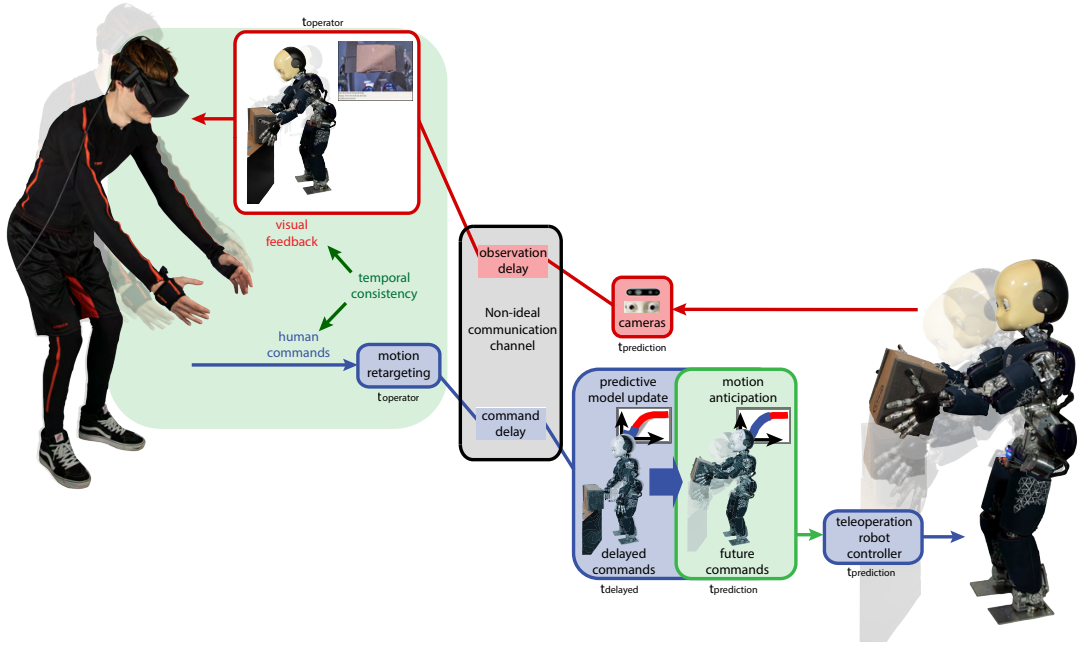


Figure 5.1: Concept of prescient teleoperation. The operator wears a motion capture suit and a virtual reality headset. They teleoperate a humanoid robot over a network with communication delays (up to 2 seconds). To send a synchronized visual feedback to the operator, the robot *anticipates* the commands thanks to data-driven predictors that are trained from a few examples beforehand: the robot executes commands that the operator has not given yet by predicting the most likely commands in the next few seconds.

to be coped with by some local robot autonomy. Hence, these techniques just offer approximate cues until the actual feedback information is available, and should be considered as support tools rather than standalone solutions.

Here, we introduce a novel teleoperation system in which the operator gets a synchronized video feed of *real images*, even when the communication channel imposes a 1 to 2 seconds delay.

Our key idea is that if the robot executes the desired movement *before* the operator performs it, then the operator will watch a delayed video feed that will be almost indistinguishable from a real-time feed (Figure 5.1). At each time-step, the robot analyzes the data that it has received so far, measures the communication time, estimates the communication time to send the feedback, and predicts what the operator is most likely to do in the next seconds. This prediction makes it possible to execute the command with enough anticipation so that the user receives a video feed that correspond to the past of the robot, but that matches the present time for the operator. We call this prediction-based feedback scheme *prescient teleoperation*.

5.2 Overview of the prescient teleoperation system

The whole-body motion of the operator is captured with a motion capture suit that computes both the joint angles and the Cartesian positions of the operator (Section 2.2). This motion is retargeted to the humanoid robot iCub. The whole-body motion of the robot is defined by the trajectories of the center of mass ground projection, the waist height, the hands positions, and the posture of the arms, of the neck and of the torso. Each sample of each of these trajectories

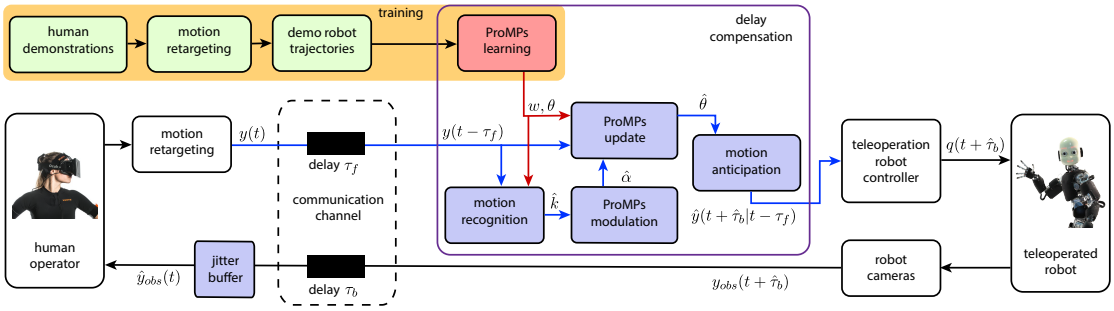


Figure 5.2: Flowchart of the proposed teleoperation system. During the training phase, the human operator teleoperates the robot without any significant delay (local network), and performs a variety of tasks. The retargeted human motions represent the robot trajectories demonstrations that are later used to train the ProMPs. A ProMP is learned for every task. When teleoperating the robot in a non-ideal network, the ProMPs are used to predict the robot movement: (1) the system recognizes the current task, that is it identifies the most likely ProMP; (2) it estimates the speed of the teleoperated motion and updates the selected ProMPs according to the delayed observed via-points; (3) it compensates for the delay by selecting the trajectory of the posterior ProMPs at the right timestep, so as to synchronize the user movements with what they see from the remote cameras at the robot side. The resulting trajectories are then tracked by the whole-body controller that computes the joint commands for the robot.

represents a control reference. Each control reference has a different priority which determines how the robot's controller computes the motors' commands and consequently how it executes the entire movement. The exact hard and soft priorities were found with a multi-objective stochastic optimizer (Chapter 3) so that the robot is unlikely to fall but tracks the trajectories as precisely as possible.

At each time-step (100 Hz), the robot searches for the joint positions by solving the redundant inverse kinematic formulated as a hierarchical constrained quadratic problem with inequality and equality constraints [59, 155] (see Chapter 3) that minimizes the tracking error, i.e., the distance to the references, while taking into account the priorities and the constraints (kinematic model, joint velocities and zero moment point bounds). In absence of delays, when prediction is not necessary, the references are the trajectories retargeted from the human operator (Chapter 2).

To operate with delays, the system predicts at each time-step the most likely future trajectory of the most likely task given the commands received so far from the operator (for instance, the 3D trajectories of the hands, etc.). The system is trained beforehand on a few example trajectories that encode the different ways of executing a task, using the principle of "motion primitives": for instance, the human is likely to reach for a bottle on the table with a similar straight hand trajectory at all times, but the trajectory may be more or less curved in presence of an obstacle. Though human gestures are generally stereotyped, the intrinsic motor noise, the human preference of movement and the small differences in the task executed in the real world (e.g., a displacement of a target object) induce variability in the human motion trajectories to realize a specific task. Even if the trajectory asked by the operator is different from the training set but included in the distributions of what has been previously demonstrated, accurate predictions can still be generated on a short time-scale.

Numerous generic machine learning techniques have been proposed to predict the future of time-series, especially with neural networks [110, 138]. Nevertheless, the robotics community has

been working for a long time on regression techniques that are well-suited for robot trajectories. In particular, Movement Primitives (MPs) are a well-established approach for representing and learning motion trajectories in robotics. They provide a data-driven representation of movements and support generalization to novel situations, temporal modulation and sequencing of primitives. Several kinds of MPs have been proposed in the literature. Dynamic Movement Primitives (DMPs) [162] are a formulation of movement primitives with autonomous nonlinear differential equations. The linear parameterization of DMPs makes them suitable for supervised learning from demonstration. Moreover, the temporal, scale, and translation invariance of the differential equations with respect to these parameters provides a useful means for movement recognition.

Our system is based on Probabilistic Movement Primitives (ProMPs) [137], which extend the concept of dynamic movement primitives to model the variance of the demonstrations. Hence, a ProMP is a distribution over trajectories, which makes it ideal tool to model the variability of human demonstrations in representing motion primitives [137]. This represents an advantage over deterministic approaches, that can only represent the mean solution.

Another advantage of working with distributions is that the properties of motion primitives can be translated into operations from probability theory. For example, modulation of a movement to a novel target can be realized by *conditioning* on the desired target's positions or partial trajectories [117]. Specifically, our system uses the ProMP's conditioning operator to adapt the predictions according to the incoming observations, hence obtaining at each timestep an updated prediction (posterior) for the prosecution of the current movement given the learned model of its associated primitive (prior). In other words, a ProMP predicts the mean trajectory of prior demonstrations when there is no conditioning data, but when some data is available, such as observations of the current movement, it adapts its prediction to resemble the most likely trajectories from the training set.

5.3 Methods

Our *prescient* whole-body teleoperation system relies on the following components (Figure 5.2):

- a whole-body controller based on quadratic optimization;
- a dataset of whole-body trajectories retargeted from human motions;
- a set of ProMPs that can predict future trajectories given observations;
- a computation of the round-trip delay to select the appropriate commands from the prediction, so that the robot anticipates both the operator-to-robot and the robot-to-operator delays;
- a blending to keep the trajectory smooth, at the start of a trajectory or in case of changes of delays;
- a video streaming system that uses a jitter buffer to cope with the stochastic part of the backward delay.

5.3.1 Training and test sets

To train our system, an operator teleoperated the iCub humanoid robot in a local network, which we consider as an approximation of an ideal network without any delay, and performed several tasks (Figure 5.3). They were wearing an inertial motion capture suit to get motion references and a virtual reality headset to get the visual feedback (Section 1.4). For each motion, we recorded

Table 5.1: Datasets used to train and test the approach. Dataset Multiple Tasks has been used to train the approach and to perform the experiments on the real robot. The testing motions represent 9 different tasks and the repetitions of each individual task share the same goal but exhibit some movement variability with respect to those used for training. Dataset Obstacles has been used to train and test the approach in simulation with respect to the presence of unexpected obstacles. The testing scenarios have obstacles in between the robot and the bottle that were not considered during training. Dataset Goals has been used to train and test the approach in simulation with respect to unexpected changes of the goal position. The testing scenarios consider bottle locations different from those included in the training.

Dataset Multiple Tasks								
		Bottle reaching						
	TOTAL	Bottle on table			Bottle on box			
#training motions	12	6			6			
#testing motions	20	10			10			
		Box handling						
		Picking up box			Placing box			
	TOTAL	low position	mid height	table	floor	table	inside box	operator's hands
#training motions	42	6	6	6	6	6	6	6
#testing motions	21	3	3	3	3	3	3	3
Dataset Obstacles								
	TOTAL	Bottle on table with different obstacles						
#training motions	6	6						
#testing motions	9	9						
Dataset Goals								
	TOTAL	Bottle on table at different positions						
#training motions	7	7						
#testing motions	10	10						

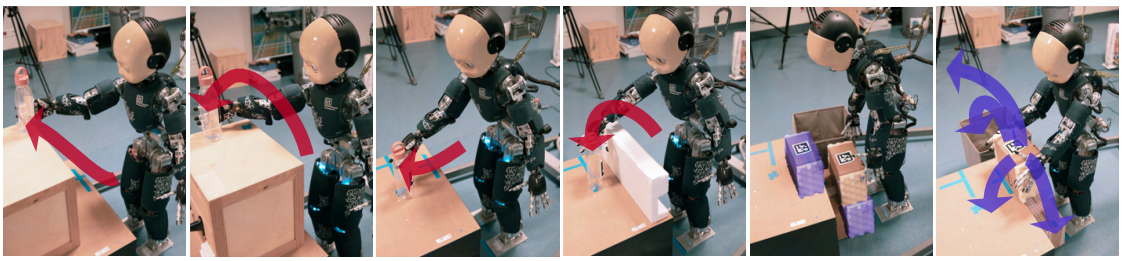


Figure 5.3: Tasks performed by the robot during the experiments (dataset "Multiple tasks"). In the first scenario (4 left images), the robot is teleoperated to reach a bottle at different locations and in different ways; in the second scenario (2 right images), the robot has to pick up a box from different locations then placing it in another location.

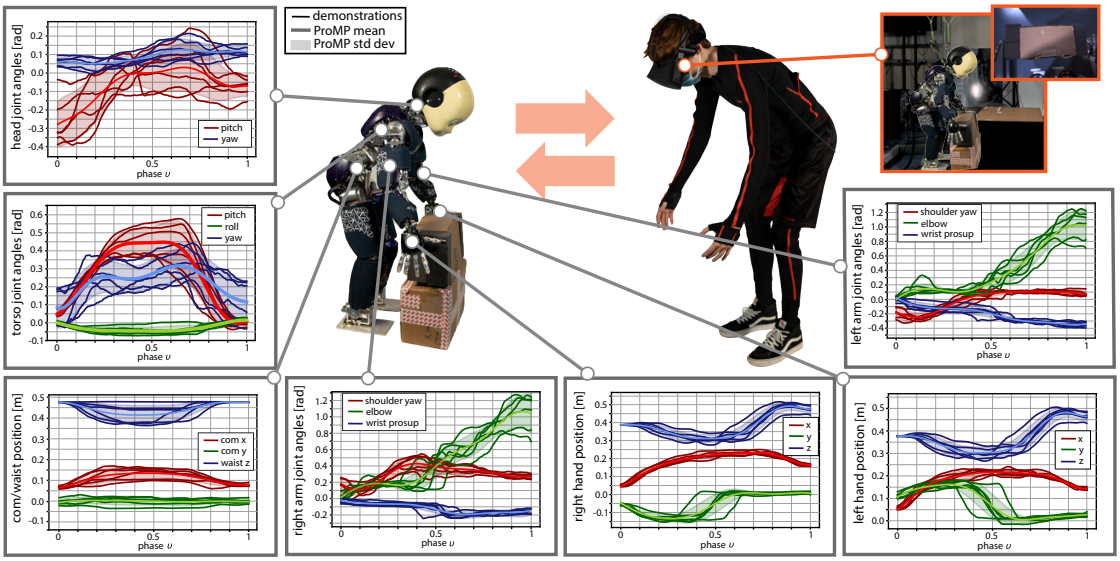


Figure 5.4: Learned ProMPs for the task of picking up a box at a low position (dataset Multiple Tasks). For each of the 6 demonstrations, the motion of the operator is first "retargeted" to the robot using the whole-body controller (ignoring delays). The trajectories of each body/joint of the robot is then recorded. From this set of demonstrations (thin lines), a ProMP is fitted for each trajectory; this ProMP is represented here as a thick line (the mean) and a light zone (the standard deviation). The computed mean is a smooth trajectory that averages all the demonstrations and the standard deviation captures the variability of the demonstrations.

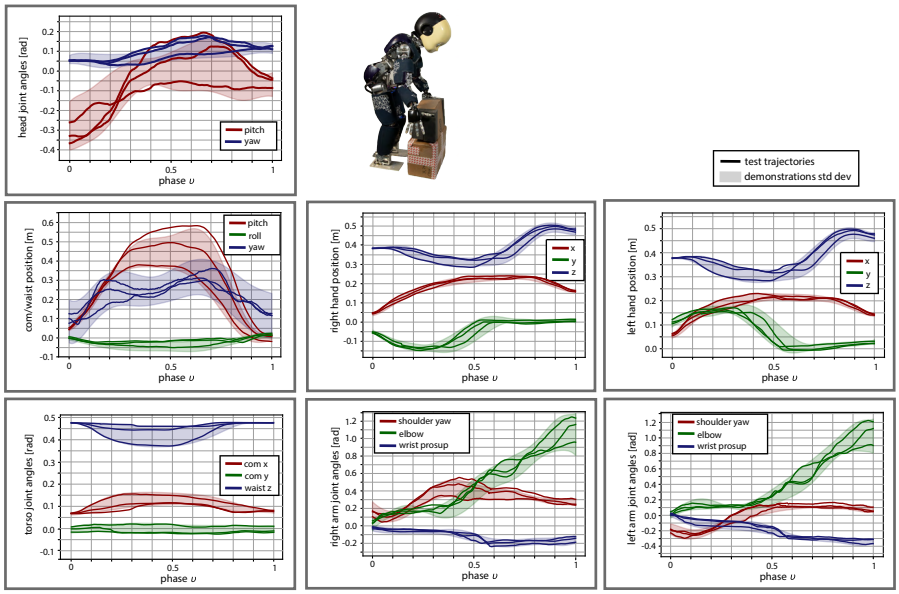


Figure 5.5: Test trajectories for the task of picking up a box at a low position (dataset Multiple Tasks). The test trajectories are different and additional repetitions of the training motions. For the tasks of picking up a box 3 different repetitions were recorded.

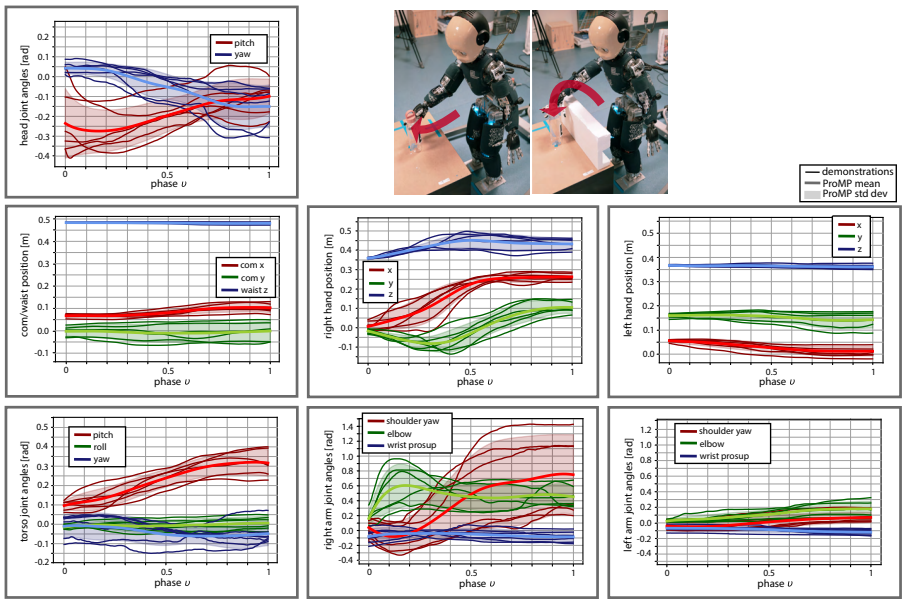


Figure 5.6: Learned ProMPs for the task of reaching a bottle on the table (dataset Multiple Tasks). The whole-body motion of the teleoperated robot is obtained by following the reference trajectories retargeted from the human. We learned a ProMP for each of these trajectories, given 6 demonstrations in a local network without any delay (3 with an obstacle in between the robot and the bottle, and 3 without)

.

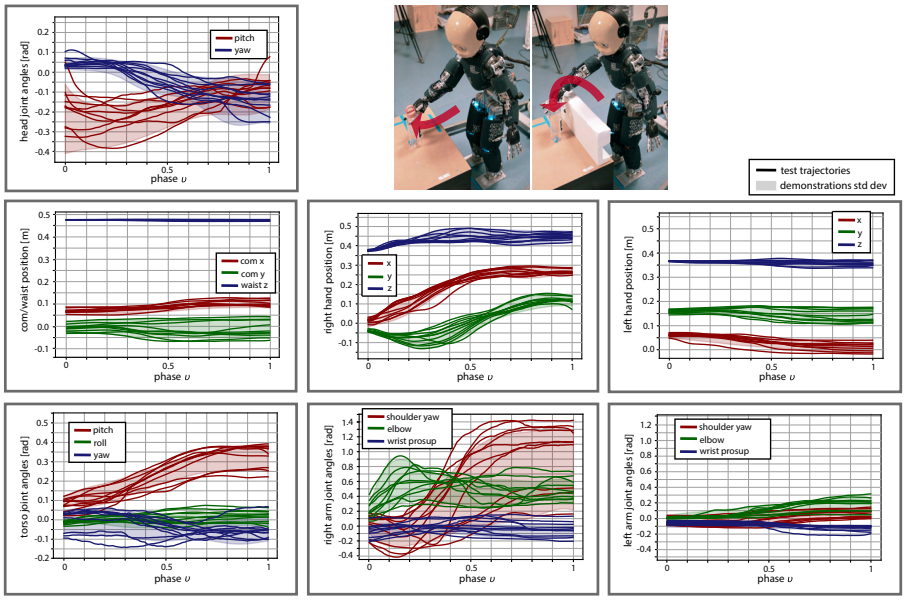


Figure 5.7: Test trajectories for the task of reaching a bottle on the table. The test trajectories are different and additional repetitions of the training motions. For the task of reaching a bottle 10 different repetitions were recorded (5 with an obstacle in between the robot and the bottle, and 5 without).

the trajectories of the center of mass ground projection, the waist height, the hands positions, the arms posture (shoulder rotation, elbow flexion, forearm rotation), the neck posture (flexion and rotation) and the torso posture (flexion, rotation and abduction). We recorded the retargeted trajectories, that is the reference trajectories for the whole-body controller of the robot. For each of these trajectories, a ProMP was learned (Figure 5.4). Hence, for each task, we have a set of ProMPs associated to the whole-body motion.

We considered several scenarios, which resulted in the acquisition of three different datasets (Table 5.1), each one divided into a training set and a test set. The first dataset (Dataset Multiple Tasks) is designed to test how well the robot recognizes tasks and deals with the intrinsic variability of the operator's movements. This is the dataset used to perform the experiments on the real robot. The second one (Dataset Obstacles) is designed to evaluate the approach with unexpected obstacles. The third dataset (Dataset Goals) is designed to evaluate the approach with novel goal positions.

Dataset Multiple Tasks. This dataset covers two scenarios (Figure 5.3): reaching a bottle with the right hand and handling a box.

The bottle task consists of demonstrations of 2 distinct whole-body reaching primitives: one primitive is for reaching the bottle on the table, the other one is for reaching the bottle on the top of the box (Figure 5.3). For each primitive, we recorded 6 repetitions of the task for training, for a total of 12 training whole-body demonstrations with an average duration of 6.1s. Every demonstration provided by the human operator is different, since it is not possible to exactly reproduce the same whole-body movement twice; to further increase the variance of the demonstrated movements, in 3 repetitions out of the 6, an obstacle was placed in between the robot and the bottle (Figure 5.3). To assess the quality of the predictions, 10 different testing repetitions were recorded for each of the two primitives; 5 with the obstacle in between the robot and the bottle, and 5 without any obstacle (Figure 5.3), for a total of 20 motions. In this dataset, the obstacles are at the same positions in both the training set and the testing set (see the dataset "Obstacles" below).

The second scenario consists of demonstrations of 7 distinct whole-body box handling primitives: 3 for picking up the box — from a low position, from a mid-height and from the table; 4 for placing the box at a specific location — on the floor, on the table, inside a container, or in a person's hands (Figure 5.3). For each primitive, we recorded 6 different repetitions for training, for a total of 42 training whole-body demonstrations, with an average duration of 7.2 s for the pick-up motions and of 5.8 s for the box-placing motions. For the test set, 3 new different repetitions of the 7 motions were recorded (Figure 5.5), for a total of 21 testing motions. Examples of training motions are showed in Figure 5.6 and Figure 5.4, while the corresponding testing motions are illustrated in Figure 5.7 and Figure 5.5, respectively.

Dataset Obstacles. The training set is composed of 6 demonstrations of bottle reaching motions with 3 different locations of an obstacle (Figure 5.8): 2 repetitions without obstacles, 2 with an obstacle in between the robot and the bottle, and 2 with a different obstacle. The average duration of the demonstrations is around 6.9s. The test set consists of motions for the same task but with obstacles at different locations (Figure 5.9): 3 repetitions for each of the 3 distinct scenarios with different obstacles (Figure 5.9).

Dataset Goals. The training set is composed of 7 demonstrations of bottle reaching motions (Figure 5.10), with the goal located in 7 different positions. The average duration of the demonstrations is 6.1s. The test set consists of motions reaching the same bottle but at 10 different

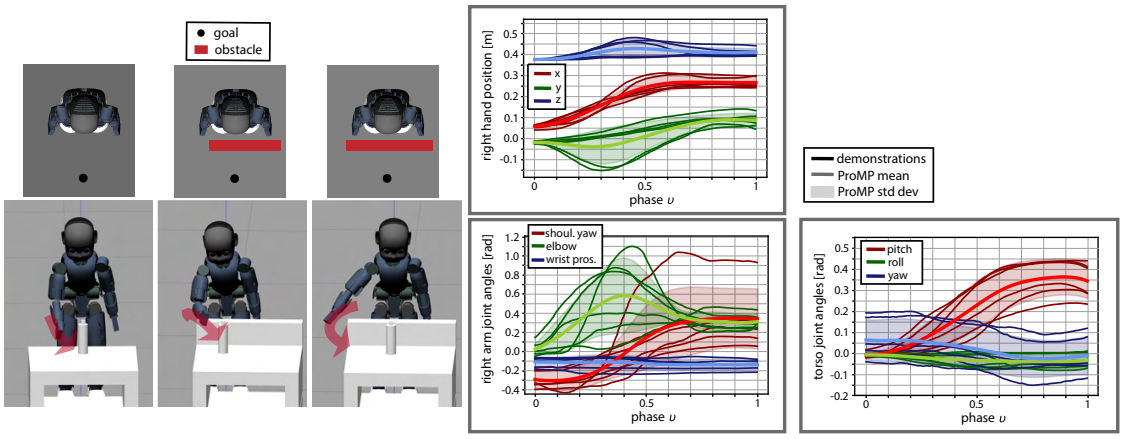


Figure 5.8: Learned ProMPs for the dataset Obstacles. The most relevant learned ProMPs and the associated demonstrations are reported. The 6 demonstrations (2 without obstacles, 2 with an obstacle in between the robot and the bottle and other 2 with a different obstacle) have been recorded while teleoperating the robot in simulation, in a local network without any delay.

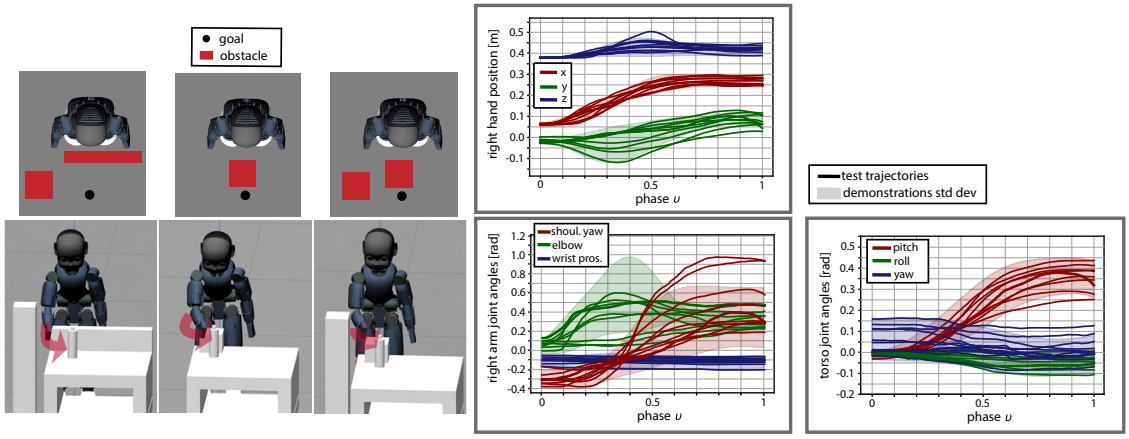


Figure 5.9: Test trajectories for the dataset Obstacles. The 9 test trajectories are different from those used for training (Figure 5.8), and consist of 3 repetitions of the bottle reaching motion for each of the 3 distinct simulated scenarios with different obstacles, illustrated on the left.

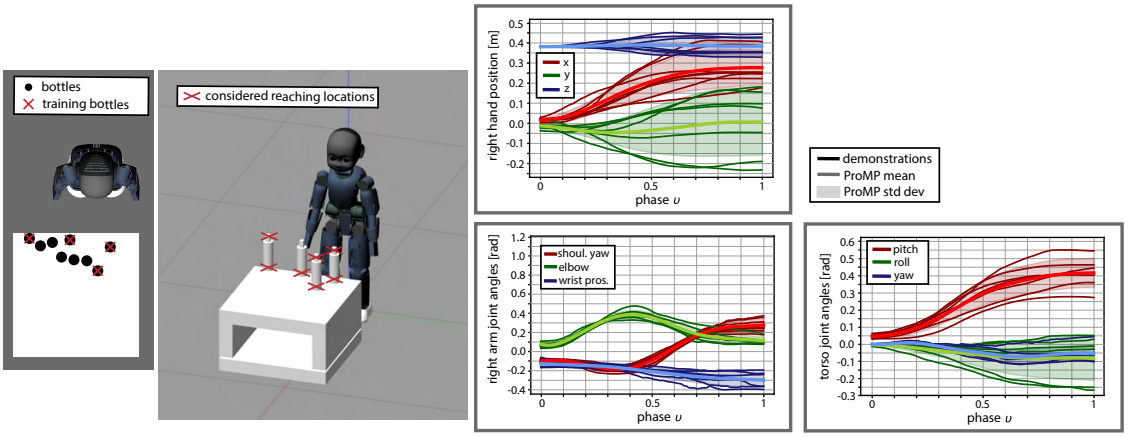


Figure 5.10: Learned ProMPs for the dataset Goals. The most relevant learned ProMPs and the associated demonstrations are reported. The 7 demonstrations have been recorded while teleoperating the robot in simulation, in a local network without any delay. The different bottle locations are illustrated on the left.

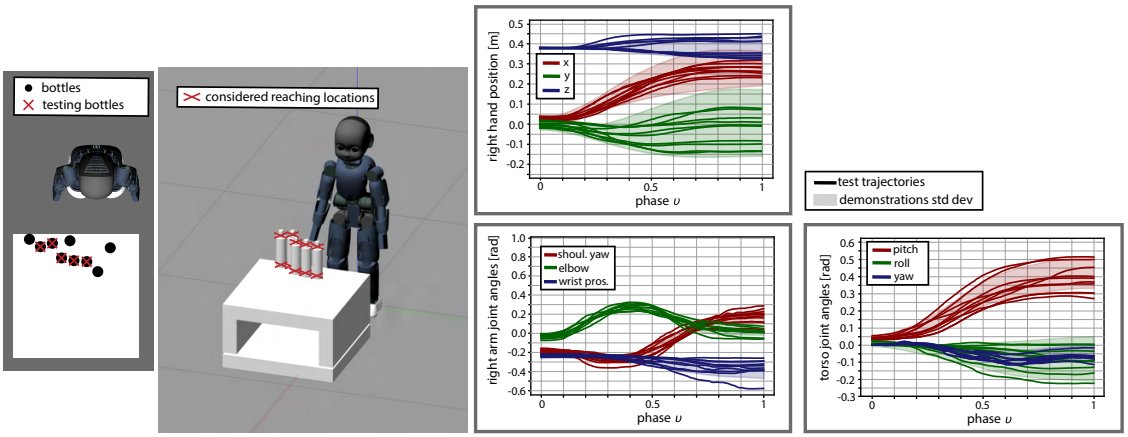


Figure 5.11: Test trajectories of dataset Goals. The 10 test trajectories are different from those used for training (Figure 5.10). The different bottle locations are illustrated on the left.

locations (Figure 5.11).

To train our system, an operator teleoperated the iCub humanoid robot in a local network, which we consider as an approximation of an ideal network without any delay, and performed several tasks (Figs. 5.3, 5.8, 5.10). They were wearing an inertial motion capture suit to get motion references and a virtual reality headset to get the visual feedback. For each motion, we recorded the trajectories of the center of mass ground projection, the waist height, the hands positions, the arms posture (shoulder rotation, elbow flexion, forearm rotation), the neck posture (flexion and rotation) and the torso posture (flexion, rotation and abduction). For each of these trajectories, a ProMP was learned (Figure 5.4). Hence, for each task, we have a set of ProMPs associated to the whole-body motion.

We considered several scenarios, which resulted in the acquisition of three different datasets, each consisting of a training and a testing set. In the first dataset (Multiple Tasks, Figure 5.3), the tasks correspond to reaching a bottle (scenario 1) or picking a box with the two hands (scenario 2). The test trajectories correspond to different repetitions of the tasks used for training, and encode the intrinsic movement variability of the operator. In the second dataset (Obstacles, Figure 5.8-5.9), the test trajectories correspond to different ways of reaching a bottle while avoiding obstacles that were not considered during training, that is, performing the same task but in different ways. In the last dataset (Goals, Figure 5.10-5.11), each motion correspond to reaching a bottle at different positions, and the test trajectories are for positions that were not used for training.

5.3.2 Whole-body Controller

The whole-body motion of the robot is defined by the following trajectories: center of mass ground projection, waist height, hands positions, arms postures, neck posture, torso posture, which are either given by the delayed retargeted human motion, or generated by the delay compensation algorithm during the execution of the main task. Each sample of each of these trajectories represents a control reference $\hat{\boldsymbol{y}}$. Given $\hat{\boldsymbol{y}}$, the robot commands $\boldsymbol{q}$ are generated by solving the redundant inverse kinematics, which can be formulated as a constrained quadratic programming problem with equality and inequality constraints (Section 1.5.1).

In our implementation, we considered as Cartesian tasks the center of mass ground position with relative weight $w_{com} = 1$, the hand positions with relative weight $w_{hand} = 0.73$ and the floating base height with $w_{base} = 0.85$. The postural tasks we included in the QP are the neck posture with $w_{neck} = 1$, the torso posture with $w_{torso} = 0.62$, the elbow and wrist postures with $w_{forearm} = 0.11$. We computed optimal priorities with the multi-objective stochastic optimization from Chapter 3. As equality constraints, we considered the feet poses that should maintain contact with the ground, since we only consider double support motions. The inequality constraints contain the robot joint velocity bounds and zero moment point bounds, which is constrained to stay inside the support polygon.

Our controller is based on the OpenSOT framework [155] and the qpOASES quadratic programming solver. This controller is run at 100 Hz, which is the same frequency at which the motor commands are updated.

5.3.3 Probabilistic Movement Primitives (ProMPs)

A ProMP [137] is a probabilistic model for representing a trajectory distribution. The movement primitive representation models the time-varying mean and variance of the trajectories and is based on basis functions. A single trajectory is represented by a weight vector $\boldsymbol{w} \in \mathbb{R}^m$. The probability of observing a trajectory $\boldsymbol{y}$ given the underlying weight vector is given as a linear basis function

model

$$\boldsymbol{\xi}_t = \boldsymbol{\Phi}_t \boldsymbol{w} + \boldsymbol{\epsilon}_\xi, \tag{5.1}$$

$$p(\boldsymbol{y}|\boldsymbol{w}) = \prod_t \mathcal{N}(\boldsymbol{\xi}_t|\boldsymbol{\Phi}_t \boldsymbol{w}, \boldsymbol{\Sigma}_\xi), \tag{5.2}$$

where $\boldsymbol{\Sigma}_\xi$ is the observation noise variance, $\boldsymbol{\epsilon}_\xi \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\xi)$ is the trajectory noise. The matrix $\boldsymbol{\Phi}_t \in \mathbb{R}^m$ corresponds to the m normalized radial basis functions evaluated at time t, with

$$\phi_c(t) = \frac{\exp\Big(-\frac{1}{2}\big(t - \frac{c-1}{m-1}\big)^2\Big)}{\sum_{c=1}^{m} \exp\Big(-\frac{1}{2}\big(t - \frac{c-1}{m-1}\big)^2\Big)}, \tag{5.3}$$

where the variable $c \in \{1, 2, ..., m\}$ specifies the center of each basis function. The distribution $p(\boldsymbol{w}; \boldsymbol{\theta})$ over the weight vector $\boldsymbol{w}$ is Gaussian, with parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_{\boldsymbol{w}}, \boldsymbol{\Sigma}_{\boldsymbol{w}}\}$ specifying the mean and the variance of $\boldsymbol{w}$. The trajectory distribution $p(\boldsymbol{y}; \boldsymbol{\theta})$ is obtained by marginalizing out the weight vector $\boldsymbol{w}$, i.e.

$$p(\boldsymbol{y}, \boldsymbol{\theta}) = \int p(\boldsymbol{y}|\boldsymbol{w}) p(\boldsymbol{w}; \boldsymbol{\theta}) d\boldsymbol{w}. \tag{5.4}$$

5.3.4 Learning ProMPs from demonstrations

The demonstrations are trajectories retargeted from the human. These are recorded in an "offline phase", while the user teleoperates the robot within a local network (approximately without delays) to perform a variety of tasks. Since the duration of each recorded trajectory may be different, a phase variable $\upsilon \in [0, 1]$ is introduced to decouple the movement from the time signal, obtaining a common representation in terms of primitives that is duration independent [51]. For each task, the modulated trajectories $\boldsymbol{\xi}_i(\upsilon)$ are then used to learn a ProMP. The parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w\}$ of the ProMP are estimated by using a simple maximum likelihood estimation algorithm. For each demonstration i, we compute the weights with linear ridge regression, i.e.

$$\boldsymbol{w}_i = \big(\boldsymbol{\Phi}_\upsilon^\top \boldsymbol{\Phi}_\upsilon + \lambda\big)^{-1} \boldsymbol{\Phi}_\upsilon^\top \boldsymbol{\xi}_i(\upsilon), \tag{5.5}$$

where the ridge factor λ is generally set to a very small value, typically $\lambda = 10^{-12}$ as in our case, as larger values degrade the estimation of the trajectory distribution. Assuming Normal distributions $p(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{\mu}_w, \boldsymbol{\Sigma}_w)$, the mean $\boldsymbol{\mu}_w$ and covariance $\boldsymbol{\Sigma}_w$ can be computed from the samples $\boldsymbol{w}_i$:

$$\boldsymbol{\mu}_{\boldsymbol{w}} = \frac{1}{D}\sum_{i=1}^{D} \boldsymbol{w}_i, \quad \boldsymbol{\Sigma}_{\boldsymbol{w}} = \frac{1}{D}\sum_{i=1}^{D} (\boldsymbol{w}_i - \boldsymbol{\mu}_{\boldsymbol{w}})(\boldsymbol{w}_i - \boldsymbol{\mu}_{\boldsymbol{w}})^\top, \tag{5.6}$$

where D is the number of demonstrations. Since a whole-body trajectory is represented by N trajectories (x, y, z position of the center of mass, of the hands, etc.), we learn a ProMP for each of the N trajectories. These ProMPs all together encode the same task k.

5.3.5 Recognizing the category of motion

Each learned k-th set of N ProMPs encodes different whole-body trajectories to accomplish a given task like picking up a box or squatting. To recognize to which set k the current teleoperated motion belongs to, we can minimize the distance between the first n_{obs} delayed observations and the mean of the N ProMPs of a group k, as done in [51]:

$$\hat{k} = \arg\min_{k \in [1:K]} \left[\sum_{n=1}^{N} \sum_{t \in T_{obs}} |\boldsymbol{y}_n(t - \tau_f(t)) - \Phi_{n, t-\tau_f(t)} \boldsymbol{\mu}_{n, \boldsymbol{w}_k}| \right], \tag{5.7}$$

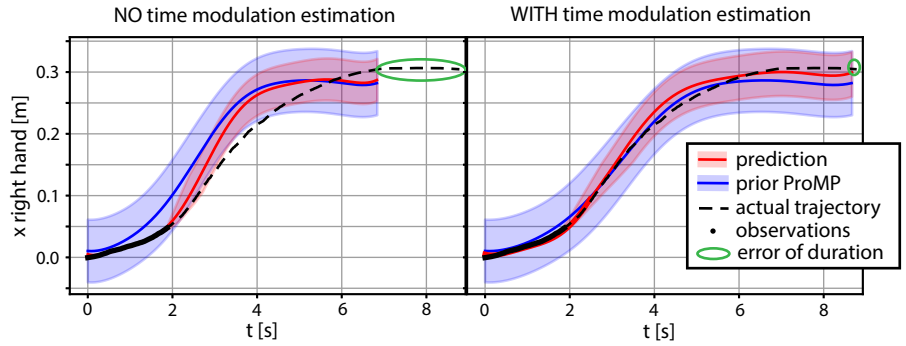


Figure 5.12: Time modulation estimation. Predicted trajectory given some early observations compared to the actual trajectory, with (right) and without (left) time modulation estimation for the task of reaching a bottle in front of the robot. Without time modulation estimation, the duration of the mean trajectory of a ProMP is set equal to the mean duration of the demonstrations, leading to mismatches between teleoperated (dashed line) and predicted (red line) motion.

where K is the number of tasks in the dataset and $T_{obs} = \{t_1, ..., t_{n_{obs}}\}$ is the set of timesteps associated to the n_{obs} early observations, and $\tau_f(t)$ the forward delay in the network. While computing $\hat{k}$, the ProMPs are modulated to have a duration equal to the mean duration of the demonstrations. The recognition (5.8) starts whenever a motion is detected, i.e. the derivative of the observed end-effector trajectories exceeds a given threshold. The distance in (5.8) is continuously computed after having identified the current motion. In this way, we can verify that the observations do not diverge from the demonstrations (exceed by a given threshold the demonstrated variance), in which case a gradual switch to delayed teleoperation is performed.

5.3.6 Time-modulation of the ProMPs

During motion recognition, we assumed that the duration of the observed trajectories is equal to the mean duration of the demonstrated trajectories, which might not be true. To match as closely as possible the exact speed at which the movement is being executed by the human operator, we have to estimate the actual trajectory duration (Figure 5.12). More specifically, we want to find the time modulation α, that maps the actual duration of a given (observed) trajectory to the mean duration of the associated demonstrated trajectories.

During the learning step, for each k-th set of ProMPs we record the set of α parameters associated to the demonstrations: $S_{\alpha k} = \{\alpha_1, ..., \alpha_n\}$. Then, from this set, we can estimate which α better fits the current movement speed. We considered the best $\hat{\alpha}$ to be the one that minimizes the difference between the observed trajectory and the predicted trajectory for the first n_{obs} datapoints:

$$\hat{\alpha} = \arg\min_{\alpha \in S_{\alpha\hat{k}}} \left\{ \sum_{t \in T_{obs}} |\boldsymbol{y}(t - \tau_f(t)) - \Phi_{\alpha(t-\tau_f(t))} \boldsymbol{\mu}_{\boldsymbol{w}_{\hat{k}}}| \right\}. \tag{5.8}$$

5.3.7 Updating the posterior distribution of the ProMPs

Once the $\hat{k}$-th most likely set of ProMPs and their duration have been identified, we continuously update their posterior distribution to take into account the observations that arrive at the robot side. Each ProMP has to be conditioned to reach a certain observed state $\boldsymbol{y}_t^*$. The conditioning for a given observation $\boldsymbol{x}_t^* = \{\boldsymbol{y}_t^*, \boldsymbol{\Sigma}_y^*\}$ (with $\boldsymbol{\Sigma}_y^*$ being the accuracy of the desired observation) is

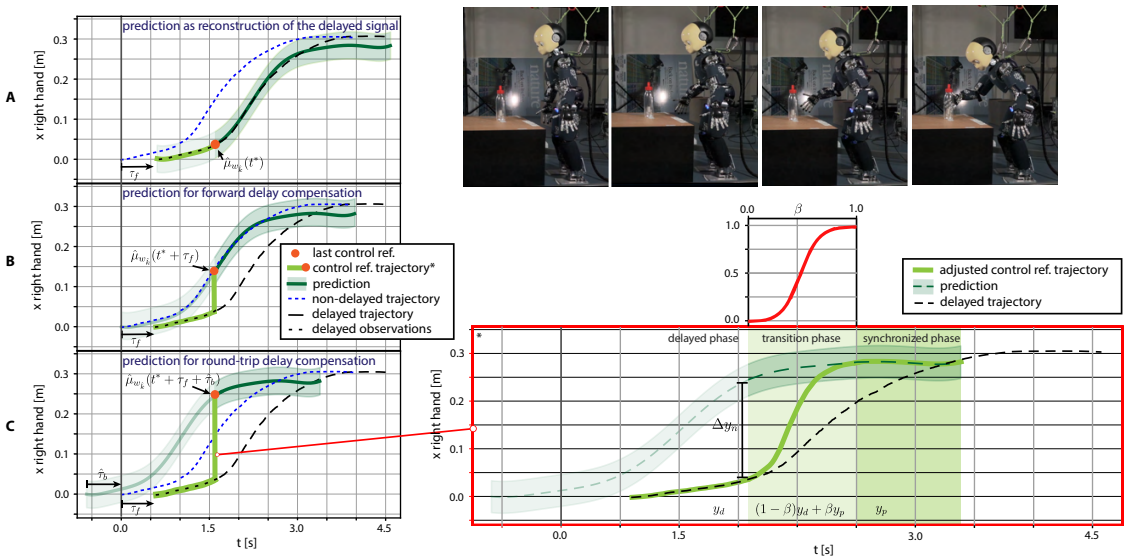


Figure 5.13: Round-trip delay compensation. Given the past delayed observations, the robot produces at each time a prediction ($\hat{\mu}_{w_k}$) of the current command. To compensate for the delays, the right sample (orange dot) from the prediction has to be selected as reference for the robot controller at each time. (A) The sample corresponding to the last received observation is an estimate of the delayed command. (B) By knowing the forward delay $\tau_f(t)$, a sample from the prediction can be selected so as to achieve a synchronization between the operator's movement and the robot movements. (C) By knowing the forward $\tau_f(t)$ and backward delay $\tau_b(t)$, the robot can select the right sample from its prediction so as to achieve a synchronization between the operator's movement and the feedback from the robot side. A policy blending arbitrates the delayed observations with the samples selected from the prediction, which guarantees a smooth transition from delayed to compensated teleoperation.

performed by applying Bayes theorem

$$p(\boldsymbol{w}_{\hat{k}}|\boldsymbol{x}_t^*) \propto \mathcal{N}(\boldsymbol{y}_t^*|\boldsymbol{\Phi}_{\hat{\alpha}t}\boldsymbol{w}_{\hat{k}}, \boldsymbol{\Sigma}_y^*)p(\boldsymbol{w}_{\hat{k}}). \tag{5.9}$$

The conditional distribution of $p(\boldsymbol{w}_{\hat{k}}|\boldsymbol{x}_t^*)$ is Gaussian with mean and variance

$$\hat{\boldsymbol{\mu}}_{\boldsymbol{w}_{\hat{k}}} = \boldsymbol{\mu}_{\boldsymbol{w}_{\hat{\boldsymbol{k}}}} + \boldsymbol{L}\big(\boldsymbol{y}_t^* - \boldsymbol{\Phi}_{\hat{\alpha}t}^\top \boldsymbol{\mu}_{\boldsymbol{w}_{\hat{\boldsymbol{k}}}}\big), \tag{5.10}$$

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{w}_{\hat{k}}} = \boldsymbol{\Sigma}_{\boldsymbol{w}_{\hat{\boldsymbol{k}}}} - \boldsymbol{L}\boldsymbol{\Phi}_{\hat{\alpha}t}^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_{\hat{\boldsymbol{k}}}}, \tag{5.11}$$

where

$$\boldsymbol{L} = \boldsymbol{\Sigma}_{\boldsymbol{w}_{\hat{k}}} \boldsymbol{\Phi}_{\hat{\alpha}t}\big(\boldsymbol{\Sigma}_y^* + \boldsymbol{\Phi}_{\hat{\alpha}t}^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_{\hat{k}}} \boldsymbol{\Phi}_{\hat{\alpha}t}\big)^{-1}. \tag{5.12}$$

Given the delay in the transmitted data $\tau_f(t)$, we can compute the timestep t^* at which the ProMP has to be conditioned to a certain observation $\boldsymbol{x}_t^*$:

$$t^* = t - \tau_f(t) - t_0. \tag{5.13}$$

where t_0 is the starting time of the current motion.

5.3.8 Delay generation

The round-trip delay $\tau(t)$ at time-step t is divided into a forward $\tau_f(t)$ (operator to robot) and a backward delay $\tau_b(t)$ (robot to operator):

$$\tau(t) = \tau_f(t) + \tau_b(t)$$

Each one-way delay is composed of two parts, one deterministic and one stochastic [70]. The deterministic component corresponds to the transmission and propagation delays. It does not change when all the transmitted packets have the same format and size and use same physical link [70]. The stochastic part, often called the "jitter" is mainly associated with the queueing delay [65] and varies from packet to packet, even when the packets have the same size and format.

If we denote by $\tau_{f,D}$ the deterministic part of τ_f and by $\tau_{f,S}$ the stochastic part:

$$\tau_f(t) = \tau_{f,D} + \tau_{f,S} \tag{5.14}$$

$$\tau_b(t) = \tau_{b,D} + \tau_{b,S} \tag{5.15}$$

In our experiments, we generate a forward delay that follows a normal distribution:

$$\tau_f(t) = \tau_{f,D} + \mathcal{N}\big(0, \sigma_{\tau_f}\big)$$

For both simulations and real experiments, we set the deterministic part of the forward delay $\tau_{f,D}$ between 100ms to 1s (depending on the experiment, see the captions of each figure) and the jitter σ_{τ_f} to $\frac{2}{15}\tau_{f,D}$, which is in line with what the jitter usually represents [23].

$$\tau_{f,D} \in [100, 1000] \text{ (depending on the experiment)} \tag{5.16}$$

$$\sigma_{\tau_f} = \frac{2}{15}\tau_{f,D} \tag{5.17}$$

For the stochastic part of the backward delay, we assume that the robot uses a video streaming software that implements a jitter buffer (sections "Delay estimation by the robot" and "Jitter buffer"), which is the case of all the modern video streaming systems. If we set the jitter buffer length to d, then this buffer adds an additional deterministic delay of d: all the packets that arrive before d seconds are re-ordered and packets that are not arrived are dropped (dropping a few frames has little consequence for a state-of-the-art video codec). As a consequence, from the operator perspective, the backward delay is constant. This is why, for both simulations and real experiments, we generate a constant backward delay:

$$\tau_b(t) = \tau_{b,D}(t) \tag{5.18}$$

For simplicity, we set $\tau_b(t) = \tau_{f,D}$ in all our experiments, but nothing in our system requires these two delays to be equal; in particular, it would be equivalent to set $\tau_b(t) = \tau_{b,D}(t) + K$ with K any constant delay caused by the jitter buffer.

5.3.9 Delay estimation by the robot

We assume that the clocks of the robot and of the computer of the operator are synchronized. In our real experiments, we synchronize the clock using the NTP system [121], which is the standard Unix protocol for time synchronization. The two clocks typically differ from less than 1ms on a local network [187]. Alternatively, GPS receivers can provide a highly accurate and absolute clock reference with an error of a few nano-seconds [187].

We add a time-stamp to each of the packets sent by the operator, which makes it possible for the robot to compute the forward delay $\tau_f(t)$ (this includes both the deterministic and stochastic part) when a packet is received at time t :

$$\tau_f(t) = \text{clock}_{\text{robot}}(t) - \text{timestamp}_{\text{operator}}(t) \tag{5.19}$$

Please note that this does not assume that the delay follows a normal distribution. If necessary, the robot can re-order the packets according to the time-stamps to condition the trajectory predictions.

While the robot needs an estimate of the backward delay, it cannot know in advance the stochastic part before sending it. Our approach is to rely on the jitter buffer (section "Jitter buffer") implemented in the video streaming system to make the backward delay deterministic: if we set the jitter buffer length to d s on the operator receiving side, then we know that the delay caused by the jitter will be exactly equal to d.

In our experiment, we therefore assume that the backward delay is known and constant (100 ms, 250ms, etc. depending on the experiments). To keep the implementation simple and easy to reproduce, we assumed that the deterministic backward delay was always equal to the deterministic forward delay (a reasonable assumption given that the same link is used for both directions) and that the stochastic part is negligible (because we chose to not add any jitter on the backward delay in the robot experiment, see the Jitter buffer section below):

$$\tau_b(t) = \overline{\tau_f(t)} = \tau_{f,D}(t) \tag{5.20}$$

$$\tau_{b,S}(t) = 0 \tag{5.21}$$

In a deployed setup, the robot would benefit from a better estimate of the average backward delay (the deterministic part) and the length of the jitter buffer. To do so, most video streaming systems use the RTCP protocol [163] to get out-of-band statistics and control information for a video streaming session that follows the RTP protocol [135]. This data would need to be sent periodically from the operator's computer to the robot so that the robot knows both d and $\tau_{b,D}$ (which are not expected to change at high speed). Alternatively, a custom system can be designed by using time-stamps on the image packets to gather statistics about the delay.

5.3.10 Jitter buffer

The jitter buffer is the component of a video streaming system [39, 130] that re-orders packets if they are delayed by less than the length of the buffer and drops packets that are too late. Much work has been dedicated to adapt its length automatically [130]: if it is too small, then the video is jittery, but if it is too large, delays are added, which is detrimental to the user (in particular during video calls). In our system, we assume that the length is fixed and known to the robot, for simplicity.

We did not implement a jitter buffer because we wanted to avoid modifying the video streaming system: reordering or dropping packets would require a considerable expertise in the internals of both the encodings (e.g., MP4) and the video streaming software. Instead, we consider that video streaming with delay and jitter is a problem that is well solved by all the current video streaming systems, as experienced by the million of users who watch videos online on smartphones with non-ideal connections.

To summarize, from the point of view of our system, the jitter buffer results in an additional but deterministic delay. However, we assume that the robot knows the value of this additional delay as well as the deterministic part of the delay.

5.3.11 Motion anticipation

The references for the robot controller are generated at each time based on the updated ProMPs' mean trajectories $\hat{\boldsymbol{\mu}}_{\boldsymbol{w}_{\hat{k}}}$. For a given ProMP, the sample $\hat{\boldsymbol{\mu}}_{\boldsymbol{w}_{\hat{k}}}(t^*)$ corresponding to the last conditioned observation, is a reconstruction of the past retargeted human input

$$\hat{\boldsymbol{\mu}}_{\boldsymbol{w}_{\hat{k}}}(t^*) = \hat{\boldsymbol{y}}(t - \tau_f(t)). \tag{5.22}$$

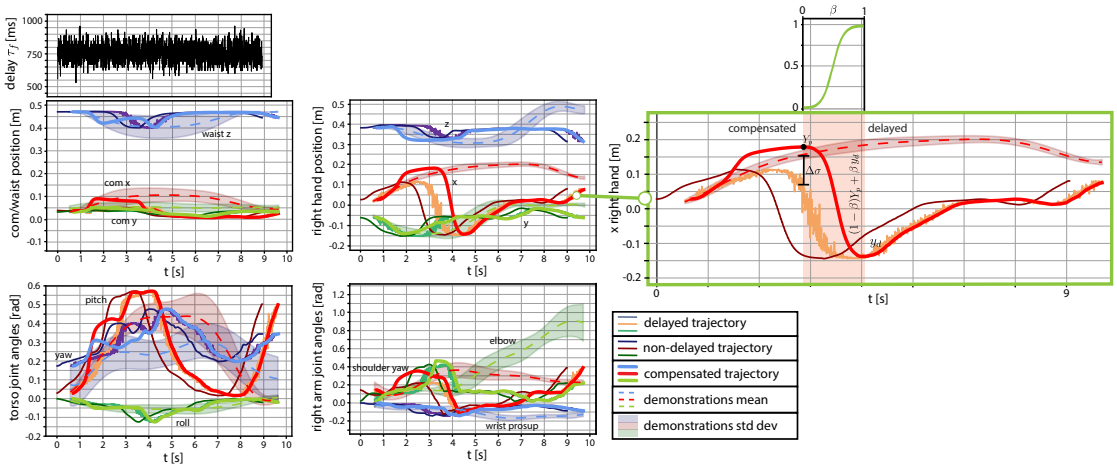


Figure 5.14: Teleoperation under unexpected human behaviors. When the last received delayed signal is too far from the learned distribution (i.e. its distance Δ_σ with the learned mean exceeds a given threshold), the prediction is smoothly blended back into the delayed signals.

The sample $\hat{\boldsymbol{\mu}}_w(t^* + \tau_f(t))$ is an estimate of the current retargeted human input

$$\hat{\boldsymbol{\mu}}_{\boldsymbol{w}_{\hat{k}}}(t^* + \tau_f(t)) = \hat{\boldsymbol{y}}(t), \tag{5.23}$$

and can be used to synchronize the human movement with that of the robot, compensating only the forward delay (see Figure 5.2). In our case, we want to synchronize the motion of the human operator with what is seen from the robot side, thus compensating for both the forward and backward delays. To do so, we select the sample $\hat{\boldsymbol{\mu}}_w(t^* + \tau_f(t) + \hat{\tau}_b(t))$ as a control reference, which corresponds to a future prediction of the retargeted human movements:

$$\hat{\boldsymbol{\mu}}_{\boldsymbol{w}_{\hat{k}}}(t^* + \tau_f(t) + \hat{\tau}_b(t)) = \hat{\boldsymbol{y}}(t + \hat{\tau}_b(t)). \tag{5.24}$$

The remaining samples $[\hat{\boldsymbol{\mu}}_{\boldsymbol{w}_{\hat{k}}}(t^* + \tau_f(t) + \hat{\tau}_b(t) + 1), \hat{\boldsymbol{\mu}}_{\boldsymbol{w}_{\hat{k}}}(t^* + \tau_f(t) + \hat{\tau}_b(t) + 2), ...]$ are also given to the controller. They are used as control references if a new reference cannot be computed in the next control step due to packet losses or jitter.

After generating a first prediction, the transition from delayed to predicted references can be discontinuous (Figure 5.13). To smoothen the transition, a policy blending arbitrates the delayed received references $\boldsymbol{y}(t - \tau_f(t))$ and the predicted ones $\hat{\boldsymbol{y}}(t + \hat{\tau}_b(t)|t - \tau_f(t))$, determining the adjusted reference (Figure 5.13):

$$\hat{\boldsymbol{y}}'(t + \hat{\tau}_b(t)|t - \tau_f) = (1 - \boldsymbol{\beta})\boldsymbol{y}_d + \boldsymbol{\beta}\boldsymbol{y}_p, \tag{5.25}$$

where $\boldsymbol{y}_d = \boldsymbol{y}(t - \tau_f(t))$, $\boldsymbol{y}_p = \hat{\boldsymbol{y}}(t + \hat{\tau}_b(t)|t - \tau_f(t))$, $\boldsymbol{\beta} = \{\beta_0, ..., \beta_n, ..., \beta_N\}^\top$ with $\beta_n \in]0, 1[$

$$\beta_n = \frac{1}{1 + e^{-12(\frac{i}{\Delta y_n} - \frac{1}{2})}}, \tag{5.26}$$

$i = \{0, 1, ..., \Delta y_n\}$ and Δy_n is the initial difference between a delayed reference and the corresponding prediction expressed in mm (for Cartesian trajectories) or $deg \times 10^{-1}$ (for postural trajectories).

5.3.12 Teleoperation under unexpected circumstances

If something unexpected happens, or if the operator suddenly changes their mind about what to do and the ongoing motion cannot be completed or is significantly altered, the prescient teleoperation is transitioned back to the delayed teleoperation. The transition from predicted to delayed

references is triggered whenever the distance between the current observation and learned mean exceeds a given threshold Δ_σ, which in the experiments was fixed equal to the learned variance plus 5cm and considered for each of the x, y, z trajectories of the hands. Since the transition can be discontinuous, a policy blending arbitrates the last predicted sample $\hat{\boldsymbol{y}}(t_{last} + \hat{\tau}_b(t_{last})|t_{last} - \tau_f(t_{last}))$ and the delayed received references $\boldsymbol{y}(t - \tau_f(t))$:

$$\hat{\boldsymbol{y}}'(t + \hat{\tau}_b(t)|t - \tau_f) = (1 - \boldsymbol{\beta})\boldsymbol{Y}_p + \boldsymbol{\beta}\boldsymbol{y}_d, \tag{5.27}$$

where $\boldsymbol{y}_d = \boldsymbol{y}(t - \tau_f(t))$, $\boldsymbol{Y}_p = \hat{\boldsymbol{y}}(t_{last} + \hat{\tau}_b(t_{last})|t_{last} - \tau_f(t_{last}))$, $\boldsymbol{\beta} = \{\beta_0, ..., \beta_n, ..., \beta_N\}^\top$ with β_n defined as in (5.26).

5.4 Results

5.4.1 Hardware and communication setup

The human motion is captured by the Xsens MVN system [156] at a frequency of 240Hz. Our compensation method receives the delayed data from the motion capture system at 100Hz and transmits to the robot controller at 50Hz.

The user teleoperating the robot is also equipped with a VR Oculus headset. Through the headset, the operator can visualize the delayed images from both an external camera at the robot side, as a third-person view of the teleoperated robot, and the robot cameras, for a first-person immersive experience. The communication protocol employed by the network is UDP with a bandwidth of 3Mbps. The forward delay is artificially generated, using the standard way to delay packets in Linux with the "netem" scheduling policy, which is based on the "iproute2" set of tools [6].

The images from the cameras at the robot side are delayed by using the open source application Kinovea [7], which allows the user to set a constant delay for the streaming of the video. The resulting delayed streaming is projected onto the VR headset through the application Virtual Desktop [13].

5.4.2 Evaluation of the predicted trajectories

Using testing data from the dataset Multiple Tasks, we evaluated the ability of conditioned ProMPs to predict the future motion of the operator, independently of any consideration for teleoperation. At each time-step, the robot identifies which ProMP best describes the current motion by selecting the ProMP that minimizes the distance between the observations so far and the mean of each ProMP, and it continuously updates the posterior distribution with the observation. Figure 5.15 illustrates the quality of the prediction for the two scenarios of the dataset Multiple Tasks in different situations due to the observed data: after observing a sufficient portion of the trajectory (around 1s) that identifies the ProMP that best describes the current motion, after a fourth of the trajectory, and after half the trajectory. The robot uses the data received so far to predict the trajectory up to the end of the experiment. Visually, the predictions match the actual trajectory very well, although it is smoother, even after less than 1s of observations. In addition, the real trajectory is almost always contained in the distribution of possible trajectories of the prediction (visualized with the variance of the ProMP). To quantitatively compare the prediction to the ground truth, we considered the whole-body motions from the testing set of the dataset Multiple Tasks and reported in Table 5.2 the corresponding prediction error between the two. The error decreases over time as more observations become available to update the prediction, reaching an error around half centimeter for the Cartesian trajectories after observing half of the motion. After observing only half

Table 5.2: Prediction error after observing different portions of the commanded trajectories (dataset Multiple Tasks). We evaluated the difference between the actual trajectory (commands retargeted from the operator) and the predicted trajectory for the 20 testing motions from the bottle reaching scenario (including those from Figure 5.7) and the 21 testing motions from the box handling scenario (including those from Figure 5.5). To understand the influence of the conditioning of the ProMPs, we computed the mean error by following the mean of the ProMP selected by hand ('no obs'), after the initial recognition ('recognition') that takes around 1s, after a fourth of the motion (1/4 motion) and after half of the motion (1/2 motion).

Trajectory	Box handling				Bottle reaching			
	RMS error [rad]				**RMS error [rad]**			
	no obs	recognition	1/4 motion	1/2 motion	no obs	recognition	1/4 motion	1/2 motion
head yaw	0.112±0.049	0.080±0.028	0.045±0.012	0.012±0.009	0.083±0.038	0.040±0.015	0.023±0.011	0.010±0.008
torso pitch	0.155±0.064	0.120±0.046	0.054±0.030	0.018±0.008	0.103±0.056	0.061±0.022	0.044±0.015	0.019±0.008
torso roll	0.119±0.049	0.103±0.038	0.055±0.028	0.017±0.008	0.082±0.042	0.054±0.016	0.032±0.010	0.016±0.008
torso yaw	0.168±0.053	0.136±0.049	0.079±0.032	0.049±0.019	0.088±0.046	0.065±0.039	0.049±0.023	0.033±0.018
r. should. yaw	0.164±0.059	0.109±0.050	0.053±0.019	0.040±0.011	0.172±0.059	0.114±0.056	0.055±0.024	0.027±0.011
r. elbow	0.169±0.072	0.146±0.038	0.097±0.032	0.033±0.010	0.220±0.083	0.097±0.032	0.065±0.017	0.034±0.011
r. wrist pros.	0.113±0.049	0.092±0.022	0.043±0.012	0.026±0.008	0.124±0.052	0.071±0.022	0.048±0.016	0.021±0.008
	RMS error [cm]				**RMS error [cm]**			
	no obs	recognition	1/4 motion	1/2 motion	no obs	recognition	1/4 motion	1/2 motion
r. hand x	3.76±0.91	2.10±0.48	1.57±0.49	0.78±0.12	3.54.±0.88	1.62±0.65	0.61±0.33	0.48±0.14
r. hand y	3.55±0.93	1.91±0.40	0.97±0.32	0.52±0.10	3.71±0.89	2.02±0.48	0.89±0.19	0.35±0.11
r. hand z	2.98±0.91	2.34±0.83	1.22±0.27	0.66±0.19	2.71±0.80	1.99±0.79	0.77±0.29	0.50±0.13
com x	2.40±0.76	0.98±0.26	0.57±0.21	0.23±0.14	0.78±0.53	0.39±0.22	0.29±0.12	0.12±0.10
com y	1.12±0.55	0.58±0.26	0.29±0.18	0.20±0.13	0.80±0.44	0.52±0.29	0.29±0.12	0.11±0.10
waist z	3.53±1.04	2.74±0.93	1.68±0.57	0.65±0.22	0.95±0.36	0.65±0.22	0.39±0.19	0.28±0.14

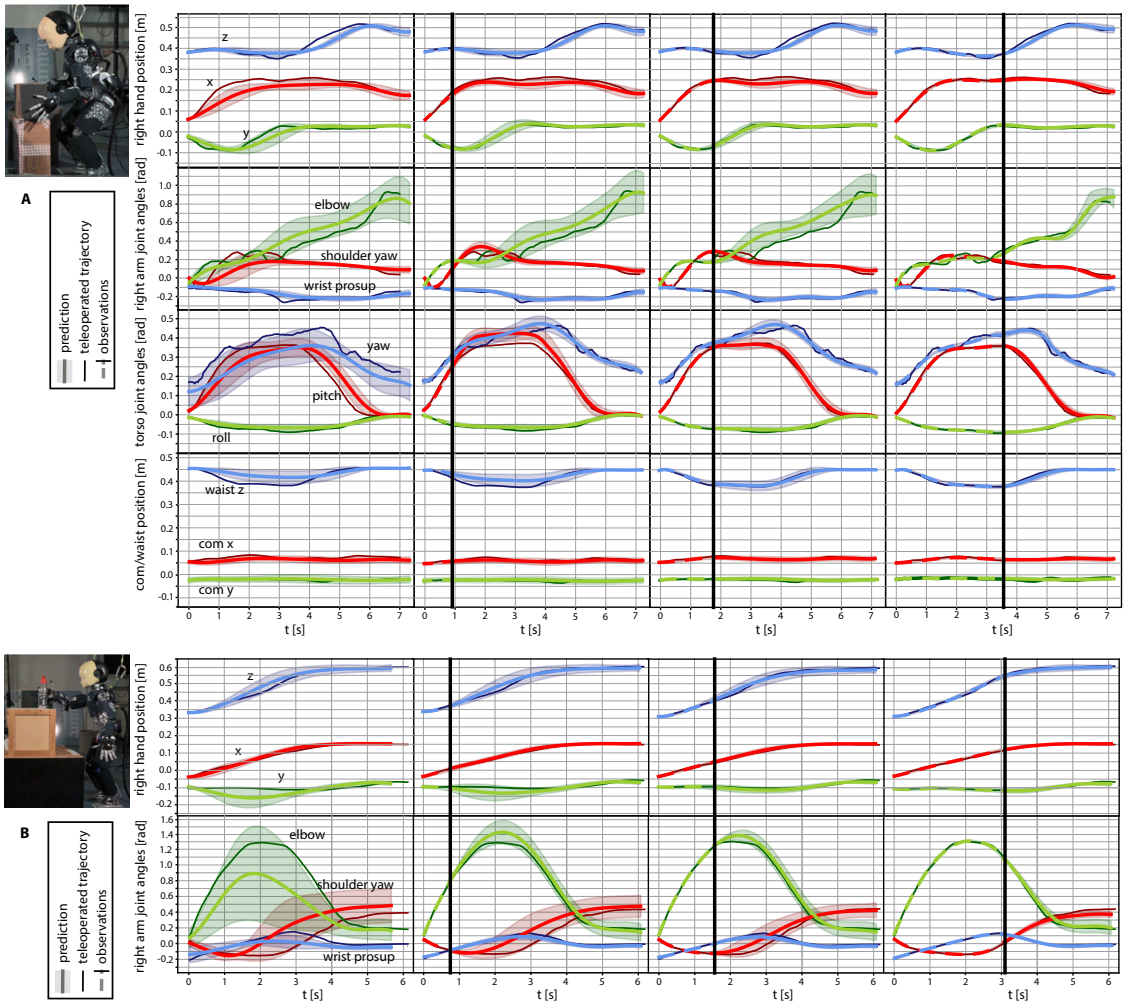


Figure 5.15: Prediction update according to observations. (A) The robot is picking up a box in front of it at a mid height. (B) The robot is reaching a bottle located on top of a box. The most relevant predicted trajectories (light colored lines) are compared to the non-delayed trajectories at the operator's side (dark colored lines), after observing different portions of the motion; a perfect prediction would mean that the light line (green/blue/red) line matches the dark line (green/blue/red). The non-delayed trajectories are from the testing scenario of dataset Multiple Tasks and the experiment corresponds to a particular case of those reported in Table 5.2. From left to right, the figure shows the prediction given by the ProMPs learned from the demonstrations, the prediction updated after observing the first portion of motion used to infer the task and its duration, the prediction updated after observing a fourth of the motion, and after observing half of the motion. After less than 1s of observation, the light trajectory is similar to the dark trajectory, whereas it might have been far from the ProMP mean (see, for instance, the elbow in experiment (b)). In most cases, 2 seconds is enough to obtain very accurate predictions for the next 4-5 seconds.

of the motion, the quality of the prediction improves by one and a half centimeters in the Cartesian trajectories of the hands with respect to the first prediction available after the recognition, which means that the ProMPs not only recognized the motion but also identified the specific execution of movement by the operator. A significant improvement can also be observed in the postural trajectories. These results show that continuously updating the prediction allows the operator to influence the predicted motion so that it matches better their intentions. Put differently, the system is not simply recognizing the motions then executing the mean of the learned demonstrations: it accurately predicts the future trajectories given the data received so far.

5.4.3 Delay generation and compensation

The time-varying round-trip delay over the network is made of a forward delay $\tau_f(t)$ in the communication from the operator to the robot, and a backward delay $\tau_b(t)$ between the robot and the operator. Each one-way delay is made of two components, one deterministic, mainly caused by the transmission and propagation time, and one stochastic [70], often called the "jitter". In our experiments, we generate forward delays with a deterministic component between 100 and 1000 ms (depending on the experiment) and a stochastic component that follows a normal distribution. We generate similar deterministic backward delays but no stochastic backward delay, because we assume that the video streaming system implements a "jitter buffer" that in effect transforms the jitter into an additional constant delay.

The robot needs to know both the forward and backward delay. It computes the forward delay exactly (both in its deterministic and stochastic components) thanks to time-stamps attached to the packets sent by the operator and synchronized clocks. The robot can easily ask the operator's computer for the average backward delay because timestamps are included in most video streaming protocols, but it cannot know the stochastic part of the backward delay before sending the packet. This is why the robot relies on the jitter buffer on the operator side to transform this stochastic delay into a deterministic delay. In our implementation, we assume that the robot knows both the deterministic backward delay (average delay) and the length of the jitter buffer (and additional constant delay). In a deployed system, these data would be gathered by the operator's computer and sent periodically to the robot.

To compensate for the delay, once the robot has identified the best ProMP that matches the observations, it has to select the predictions that correspond to the right time-step from the mean of the ProMP. By computing the round-trip delay, the robot chooses the right samples from the current movement prediction, i.e. posterior ProMPs' distributions (Figure 5.13). At the beginning of the motion, before any ProMP is recognized, the robot uses the delayed commands; however, once a ProMP is recognized, the robot can start compensating for the delays, but first the delayed trajectory needs to catch up with the ProMP. To keep the trajectory smooth, we take inspiration from the shared control literature [113] and use blending between the current trajectory and the mean of the selected ProMP.

5.4.4 Prescient teleoperation experiments

We evaluated our *prescient* whole-body teleoperation system on the iCub robot with a time-varying round-trip delay around 1.5s, given by a stochastic forward delay following a normal distribution with 750ms as mean and 100ms as standard deviation, and a constant backward delay of 750ms (Figure 5.16 and Figure A.1). We used a constant backward delay in these experiments because we relied on an existing video streaming system that cannot artificially delay images randomly. Nevertheless, when a jitter buffer is used, the resulting video feedback is delayed by a constant delay,

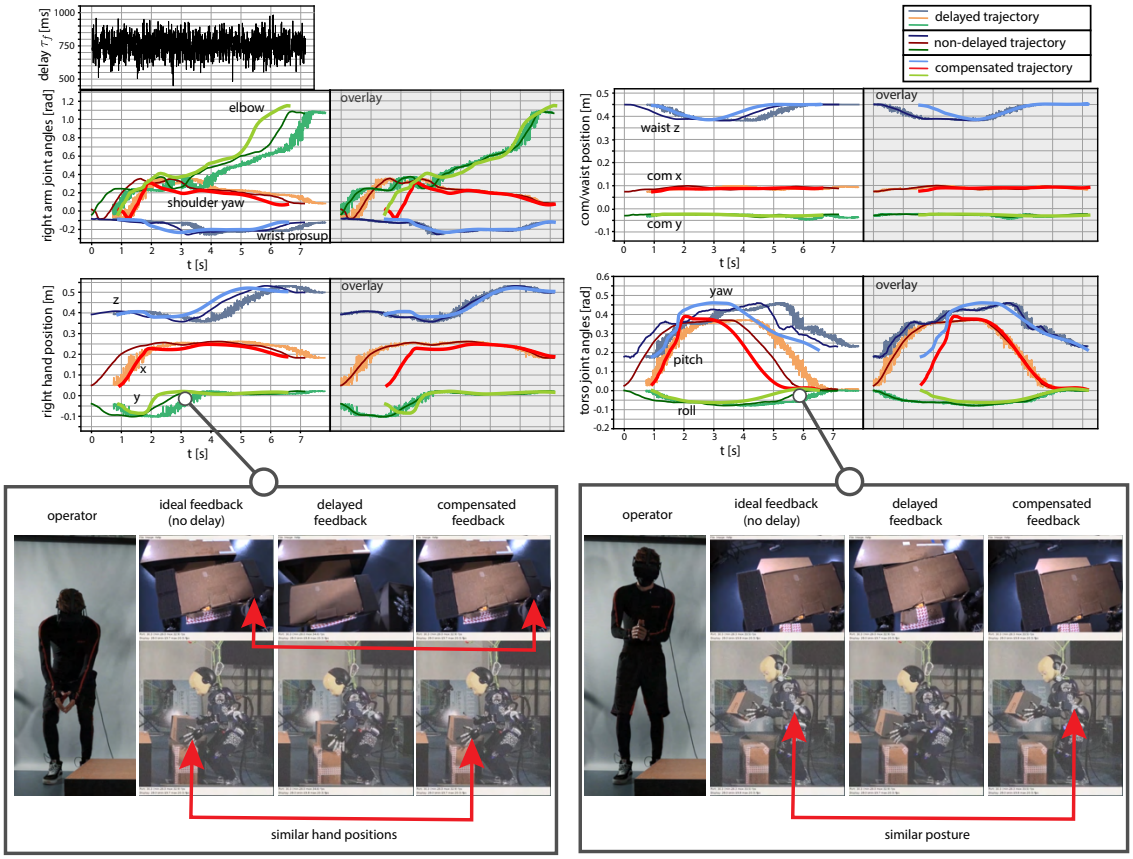


Figure 5.16: Teleoperation with compensation of a round-trip delay around 1.5s. The robot is picking up a box in front of it at mid-height. The forward delay follows a normal distribution with 750ms as mean and 100ms as standard deviation, while the backward delay is 750ms. The trajectories retargeted from the human to the robot without any delay are the dark-colored red-green-blue lines. The orange-teal-gray lines are the corresponding delayed trajectories and the light-colored red-green-blue lines are the compensated trajectories. The compensated trajectories (light red-green-blue lines) are the final robot reference trajectories. These, first follow the delayed teleoperated signals (orange-teal-grey lines). Then, when the prediction is available, they anticipate the teleoperated motion (dark-colored red-green-blue lines) so to get a visual feedback at the user side coherent with what the operator is doing.

Table 5.3: Difference (root mean square error) with the non-delayed trajectories, for both the compensated and the non-compensated (delayed) trajectories (average delay: 1.5 s). The error is computed for the 20 testing motions from the bottle reaching scenario of the dataset Multiple Tasks (including those from Figure 5.7), and for the 21 testing motions from the box handling scenario of the dataset Multiple Tasks (including those from Figure 5.5). The compensated trajectories are temporally realigned with the non-delayed trajectories for computing the error, which is considered only once the prediction starts, and once the blended transition from delay to compensation is over (Figure 5.13). The time-varying forward follows a normal distribution with 750ms as mean and 100ms as standard deviation. The backward delay is set equal to 750ms. Examples of compensated trajectories are displayed in Figure A.6.

	Box handling **RMS error [rad]**		Bottle reaching **RMS error [rad]**	
	compensation	no compensation	compensation	no compensation
head yaw	0.024±0.011	0.035±0.012	0.013±0.007	0.021±0.011
torso pitch	0.045±0.020	0.136±0.064	0.027±0.012	0.041±0.019
torso roll	0.022±0.011	0.089±0.038	0.015±0.008	0.020±0.010
torso yaw	0.069±0.028	0.129±0.055	0.019±0.009	0.032±0.011
r. shoulder yaw	0.071±0.025	0.145±0.051	0.065±0.018	0.221±0.092
r. elbow	0.062±0.020	0.171±0.067	0.096±0.030	0.194±0.071
r. wrist prosup.	0.025±0.007	0.077±0.033	0.054±0.012	0.091±0.041
	RMS error [cm]		**RMS error [cm]**	
	compensation	no compensation	compensation	no compensation
r. hand x	1.02±0.31	2.95±1.12	1.29±0.33	4.97±1.46
r. hand y	0.90±0.26	3.36±1.21	1.21±0.31	4.33±1.17
r. hand z	0.96±0.30	2.96±0.75	1.11±0.25	4.15±1.13
com x	0.90±0.13	1.24±0.36	0.33±0.07	1.01±0.20
com y	0.79±0.11	1.06±0.39	0.24±0.06	1.01±0.32
waist z	0.88±0.14	2.02±1.22	0.22±0.07	0.61±0.09

hence our implementation has not affected the final outcome of the experiments. Additional experiments are presented in Supplementary Materials, in which different tasks are performed under stochastic round-trip delays ranging from 200ms to 2s (Figure A.2-A.5). In all our experiments, the operator was able to successfully complete the tasks in spite of these large delays (Figure 5.16) thanks to the compensation. All the experiments can be seen in the video youtu.be/N3u4ot3aIyQ.

If the operator decides to stop or to perform a different movement from the one that has begun, then the predicted trajectories are blended into the delayed trajectories in a way similar to that adopted to switch from delayed to predicted references (as shown in Figure 5.14), hence avoiding any undesired prolonged mismatch.

To quantify the quality of the compensation, we compared the compensated trajectory to the non-delayed trajectory in the 20 testing motions from the bottle reaching scenario of the dataset Multiple Tasks and for the 21 testing motions from the box handling scenario of the dataset Multiple Tasks, with time-varying forward delay following a normal distribution with 750ms as mean and 100ms as standard deviation and backward delay being equal to 750ms (Table 5.3). The comparison is performed once the prediction is available and after the transition from delayed signals to prediction. In the box handling scenario, the results show that the error is around or less than 1cm for all the considered references (in particular for the hands) whereas without compensation the error is about three times higher (about 3cm for the hands). Similarly, in the bottle reaching task, the error of the compensated trajectory is about 1 to 1.4cm for the hands versus about 4cm for the hands and 1cm for the center of mass when there is no compensation. The angular errors show a similar pattern. While an error of about 1cm is often enough to achieve a task, for instance grasping an object, an error of 3 to 4cm makes it very likely to miss the object, in addition to frustrating and disorienting the operator.

We then evaluated the performance of the compensation when the communication delay increases (Figure 5.18A). To do so, we compared the compensated trajectory to the non-delayed trajectory, for the right hand, in the task of reaching the bottle on the table of the dataset Multiple Tasks (Figure 5.7). During the synchronization, the error is roughly proportional to the delay (Figure 5.13), which adds up to the prediction errors. In that case, we observe a mean error of about 2.5cm for 1s delay, but more than 10cm for a 3s delay, because the transition time takes a significant amount of time on a short trajectory (30% for a delay of 3s and a trajectory of 10s). In longer movements or in an actual teleoperation session where the operator commands longer sequences of movements, this synchronization time should become negligible. When we exclude the synchronization time, the results show that the compensated tracking error is less than 2cm for delays around 0.5s, about 2.5cm for delays around 1.5s, and increases to about 5cm for delays of 3s and 4s. Qualitatively, the operator found the system difficult to use with a delay of more than 2s, for which the error is about 3cm after the transition and around 7cm including the transition.

5.4.5 Conforming the teleoperation to the intended motion

Figure A.6 shows that the user, and therefore the robot, can pick up a box by bending the back without using the legs, by bending the legs and limiting back movements, or any other way in between. This means that the robot is not following the mean trajectories learned from the training demonstrations, but that it adapts the teleoperation trajectories to conform with the received data. Such a feature can be obtained if the delay represents a reasonable portion of the whole motion duration, that in our experiments was a delay around 1.5s for motions with a duration around 6s/7s. When the delay increases to a significant portion of the complete motion duration, the early delayed observations may not be sufficient to update the posterior distribution of the ProMPs following the actual operator's commands (especially when the demonstrations exhibit a late variance). In this

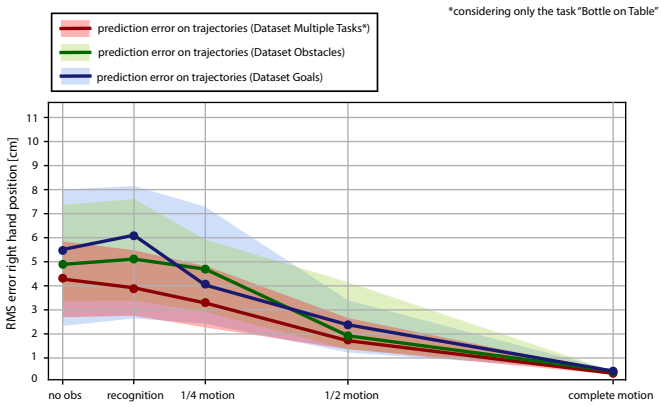


Figure 5.17: Comparison of the prediction error on different datasets regarding the task of reaching a bottle on the table. The RMS of the error is computed based on the 10 testing trajectories of the task of reaching the bottle on the table from Figure 5.7 (dataset Multiple Tasks), the testing trajectories from the dataset Obstacles (Figure 5.9), and the dataset Goals (Figure 5.11). The bold line represents the median RMS error. The transparent region around it represents the IQR of the error. The prediction error is computed as the Euclidean distance between the predicted trajectory and the reference trajectory. The plot reports the error given by the mean trajectories of the ProMPs learned from the demonstrations, from the prediction updated after observing the first portion of motion used to infer the task, after observing a fourth of the motion, after observing half of the motion, and after observing the whole motion.

case (Figure A.7), we obtain a robot behavior that tends to follow the mean of the demonstrations for a significant part of the task. This is visible in the video youtu.be/N3u4ot3aIyQ (1min53s) where, for a round-trip delay around 2s, the robot picked up the box by bending the knees while the human was only using the back without moving the legs.

To experiment explicitly with this adaptation of motion, we teleoperated the robot in simulation to reach a bottle located onto the table in 3 different ways during training (Figure 5.8, dataset "Obstacles"). During testing, we evaluated our compensation approach on the same task but avoiding different obstacles (Figure 5.9). Like with the previous dataset, the prediction continuously improves as more observations are available, trying to fit the observed operator commands (Figure 5.17-A.8): the quality of the prediction improves over time in a similar fashion to the predicted trajectories from the bottle reaching scenario in Figure 5.7, reaching about the same accuracy after observing half of the motion. To evaluate how the resulting compensation performs with different delays, we compared the compensated trajectories for the right hand position to the corresponding non-delayed ones in the 9 testing motions (Figure 5.18B). The results show that the error in position is comparable (if higher, only by about half a centimeter) to the error from the bottle reaching scenario, where the same obstacles were used for training and testing.

Some examples of the compensated trajectories are illustrated in Figure A.9, where the robot is reaching the bottle while trying to avoid new obstacles. After an initial recognition phase, the compensated trajectories follow the specific way the human is commanding the robot in order to avoid the different obstacles, which could have been hit if the mean trajectories from the demonstrations were used. For instance, if we look at the right shoulder yaw angle (top row, center, red line), we see that both the prediction and the command are far from the mean of the ProMP: when the operator approached the object by the side (see the red arrow in the picture of the robot), the system adapted to perform the motion in the same way as the human. Similar observations can be

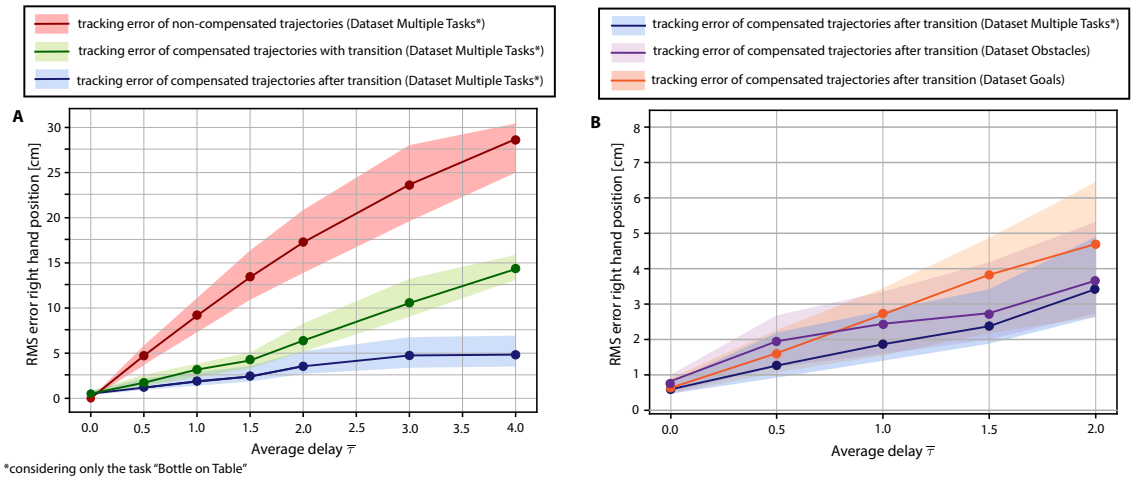


Figure 5.18: Scalability of the delay compensation with respect to increasing time delays. (A) Tracking error of the compensated trajectories for the right hand position with respect to the non-delayed ones compared to the tracking error of the corresponding non-compensated (delayed) trajectories with respect to the same non-delayed ones. The tracking error of the compensated trajectories is considered both including the transition from the delayed phase to the synchronization phase (Figure 5.13), which adds a non-compensable error, and without transition. The RMS of the error is computed from the 10 testing motions of the task of reaching the bottle on the table from dataset Multiple Tasks (Figure 5.7) and its median value is reported as a bold line. The transparent region around it, represents the IQR of the error. (B) The tracking error of the compensated trajectories for the right hand position with respect to the non-delayed ones (after the transition phase) is evaluated also on the testing trajectories from the dataset "Obstacles" with different obstacles (Figure 5.9) and the dataset Goals with different reaching goals (Figure 5.11). The tracking error is computed as the Euclidean distance between the evaluated trajectory and the reference trajectory. The compensated trajectories are temporally realigned with the non-delayed trajectories for computing the error, which is evaluated with different round-trip delays $\tau(t)$: 0s, around 0.5s, 1s, 1.5s, 2s, 3s and 4s. The time-varying forward delay follows a normal distribution with mean $\overline{\tau_f} = \overline{\tau}/2$ and standard deviation equal to $\frac{2}{15}\overline{\tau}_f$. The backward delay is set equal to $\overline{\tau}_f$.

made for the other degrees of freedom (e.g., the y position of the right hand: green line on the top left panel; the yaw of the torso: blue line on the top right panel). For reference, Figure A.10 shows how the same trajectories fit almost exactly the actual trajectories retargeted from the human in the case of no delay, which confirms that when there is no delay the robot almost perfectly follows the orders from the operator, even if they do not correspond to any training trajectory.

5.4.6 Conforming the teleoperation to new goals

Similarly to how we tested the adaptability of the approach to the specific way the operator is performing a given task, we also investigated its ability to adjust to new object locations. To do that, we recorded a third dataset (data "Goals") by teleoperating the robot in simulation to reach a bottle located onto the table in several positions (Figure 5.10), then we tested the approach while reaching the same bottle but at different locations (Figure 5.11).

As one could expect, the prediction is less accurate than with the previous datasets where the goal position is always the same (5.17-A.11). During the initial recognition phase the prediction error is around 2.5cm higher compared to the trajectories from Figure 5.7, but it continuously improves as more observations are available, decreasing the error gap to 1cm after observing half of the motion. For delays around 1.5s, the average tracking error on the hand position is 4cm (Figure 5.18B), which is about a centimeter higher than the error with a delay of 2s on the datasets in which the bottle is in the same position for both training and testing. Since the operator found it difficult to teleoperate the robot with such compensation errors, the performance of the approach on these tasks cannot be considered reliable for delays around 1.5s. However, the approach can still be used with novel goals for lower delays, typically between 0.5 and 1s, when a similar accuracy to the other datasets is often obtained (Figure 5.18B).

Some successful examples of compensated trajectories are illustrated in Figure A.12, where the robot is reaching the bottle located at unexpected positions. After an initial recognition phase, the compensated trajectories try to follow the operator's commands, which are guiding the robot hand toward the new target. The figure shows that both the prediction and the commands are far from the mean of the ProMP, which means that using the mean trajectories from the training demonstrations instead of our approach, would have certainly made the robot miss the bottle.

5.5 Discussion

Humanoids can only be deployed at their full potential if they can exploit their whole-body to perform non-trivial, high-value tasks. Whole-body teleoperation is the ideal framework to achieve this goal because it provides an intuitive and flexible approach to operate the robot, provided that the operator can rely on a synchronized feedback. By leveraging machine learning to anticipate the commands of the operator, we showed that it is possible to compensate for delays of 1-2 seconds, which typically correspond to the round-trip communication time between Earth and space [108] and between continents on the Internet [76].

To achieve the synchronization between the human motion and the visual feedback as soon as possible, our approach strongly relies on fast motion recognition. In fact, if this step were to take too long, the motion would be over by the time any compensation algorithm could be applied.

In our experiments we leveraged the ProMPs to represent human gesture primitives and also to predict the operator's intention of movement. In the past, we had already used ProMPs with success for real-time prediction of whole-body movements [51] and robot's gestures [50]. They only require a few robot/operator demonstrations to be trained, and the posterior update that en-

ables to predict the future trajectory also requires a relatively small set of observations. For the above elements, they are a suitable and valid technique for predicting the operator's intention in our teleoperation problem. Other data-driven methods for predicting gestures as time-series could be used in principle, for example LSTM-RNN [25, 41, 198], at the expense of more training data and potentially less smooth movements. Put differently, the concept of *prescient teleoperation* can be implemented with any other predictor, including neural networks.

We showed that the robot follows the specific execution of motion, i.e., the particular way the human is performing the task, by continuously updating the prediction of the current motion (thanks to the conditioning operator of the ProMPs). In the experiments, we were able reach a bottle in ways that had not been demonstrated before (but included in the distribution of the training demonstrations), avoiding new obstacles and reaching new object positions that were not included during the training phase. This feature is extremely valuable because it allows the operator to adapt the commands on the fly to situations different from those in the training. For instance, there might be an obstacle that forces the operator to approach an object with the hand from the right, and prevent them from following a straight path. Or there might be a low ceiling that constrains the operator to bend the torso more or flex the legs more. More importantly, there are many different objects and many different environments and the operator will adapt their movement to the local condition, from a different target position to a different body posture. Thanks to a diverse set of ProMPs and the "conformation" of the prediction to the observations, our system should be able to handle most of these cases. The main limitation is that our approach assumes that the received data is enough to update the predictions; this is often the case, because humans tend to anticipate their change in a trajectory, that is, the beginning of the trajectory changes when a future goal changes. Nevertheless, the system is not capable of anticipating a last-second change when the beginning of the trajectory is always the same. Future work should evaluate how many demonstrations (and how many ProMPS) are needed to adapt to as many situations as possible.

As shown in the video youtu.be/N3u4ot3aIyQ and Figure 5.16, we were able to identify the current motion with less than a second of data, time during which the delayed signals were used to teleoperate the robots. However, our experimental setup included a limited amount of possible tasks (see Figure 5.3). If the dataset of possible tasks was larger, the recognition would take a considerable and prohibitive amount of time. In this case, one could resort to other data-driven prediction methods or exploit the fact that human actions are mostly vision-guided and identify the object that is about to be manipulated. In such a way, the set of possible tasks would be reduced to those related to that object, accelerating significantly the motion recognition. This can be done by using object recognition algorithms [180] or by attaching ArUco markers [159] on the objects of interest. Then, if the number of tasks related to an object was similar to the number of tasks we considered in our experiments, the motion recognition would take an amount of time close to that of our experiments. Instead, if a considerably larger amount of tasks was related to an object, another strategy to reduce the recognition time could be to examine first the initial position of the hand trajectories in order to limit the recognition computation to those tasks having those initial positions in their distribution.

The proposed compensation algorithm also relies on the fact that the delay is not comparable to the duration of motion. Indeed, it would not be possible to apply any compensation, if the motion was over before computing any prediction. This is an additional reason for executing the teleoperated tasks at a slow pace (the humanoid robot in our experiments is never teleoperated with fast or rough movements anyway, to avoid damaging the platform). In our experiments we were able to compensate for the delays while following the way the human performs the task with delays around 1.5s. For delays around 2s, we were still able to compensate for the delay but the robot was not following the human precise movements, rather the mean of the learned motions.

For longer delays the performance deteriorates critically, producing robot behaviors that cannot adequately compensate for the delay.

In addition, the execution speed of a task has to be similar to those recorded during the demonstrations. Once estimated the duration of the current motion, our algorithm does not take into account any duration variation while carrying out the task. A real-time adaptation of the speed execution could be addressed by constantly updating the time modulation of the ProMPs based on the last observations and updating the posterior distribution of the original learned ProMPs based on these observations [51].

To better evaluate the delay compensation, some straightforward metrics or measures on what can be tolerated should be defined. The difficulty of compensating for time delays could be found based on measurable characteristics of a task or of an environment. For example, if the operator was trying to remotely balance an inverted pendulum, the tipping time constant of the pendulum would be relevant. A long pendulum on the Moon would allow for a much longer time delay than a short pendulum on the Earth.

Finally, all the experiments reported here were performed by a highly experienced user, who is very familiar with the teleoperation system and the iCub robot. While novice users would not have the same proficiency with the proposed system, they are very unlikely to be trusted to operate a highly expensive humanoid robot in a high-value mission, such as an intervention in a damaged chemical plant or in a remote Moon base. Like drone pilots who are trained extensively before their first mission, one should expect actual humanoid operators to be expert users with a long training period.

The next natural step is to include haptic feedback in addition to visual feedback. First, whole-body feedback would require the operator to wear a full-body exoskeleton [151] or to use less intuitive distributed vibrotactile feedback [32]. In bilateral systems the force signal is directly coupled between robot and human [151], while in master-slave systems the human operator can receive kinesthetic cues not directly related to the contact force being generated by the robot, or as indirect forces in a passive part of her/his body [32]. Our approach could certainly be extended to the latter case, where ProMPs could be learned for the force signals. The extension to bilateral techniques represents a more complex challenge, since these systems can be very unstable under time-delays. Several approaches have been proposed to stabilize these systems under time delays [171, 196] but the extension to platforms with a high number of degrees of freedom like humanoid robots and their robustness to packet loss and jittering is still an open challenge. A promising research avenue is to combine our approach with the one proposed in [186], in which a haptic feedback is produced using the point cloud data obtained with an RGB-D camera. So far, our robot has been streaming images, but it could very similarly stream a point a cloud. Like with images, this point cloud would correspond to past perceptions of the robot but the operator would perceive it as synchronized because the robot anticipated the commands. The haptic feedback from the point cloud would then not appear to be delayed.

In this sense, both visual and haptic feedback could be improved by using more cameras. In our experiments, we had to use an external camera (in addition to the robot cameras) to provide a better situational awareness to the operator about the robot's status in the environment. In fact with the limited field of view of the cameras in the iCub's eyes, the human can hardly see the environment and the robot's body at the same time, which makes it extremely difficult to grasp objects without making any errors. More situation awareness could also be provided to the operator by integrating our method with already existing predictive-display-based techniques [122]: during the non-compensated intervals the predictive display could guide the human operator with a gradual shift from the virtual graphics to the real images of the robot cameras once the synchronization with the user commands has been achieved by our approach.

Overall, this new approach will help deploy more robust, effective teleoperated systems. It is demonstrated here with a motion capture suit, a virtual reality headset and a state-of-the-art humanoid robot; but this is a general framework that could be used in any other robot, from manipulators to cars, as any of these could all anticipate remote commands provided that there is a good enough predictor.

6

Conclusions

6.1 Discussion

The work in this thesis aimed at developing a teleoperation system for humanoid robots. Even though our experiments were conducted exclusively on a single real robot, in principle the methods can be generalized to all humanoid robots. In Chapter 1, we first gave an overview of the general architecture shared by existing teleoperation systems. They all commonly include the perception of human states, their mapping to the robot (also called *motion retargeting*), a robot controller, and the perception of the robot states and the remote environment that represent the feedback for the operator, We presented the different measurement devices that can be used to capture the human motions, commands and/or states, together with the possible feedback interfaces that allow the human to perceive the robot states. In our implementation, we relied on an inertial technology-based motion capture system to track the human movements and used a virtual reality headset to transmit to the user the view from the cameras of the robot and other external cameras.

In Chapter 2, we presented our retargeting strategy. We tracked the human motion with the motion capture suit and processed the human information to produce equivalent behaviors on the robot in real-time. The motion capture system produces a reconstructed human skeleton. From this, we mapped the joint angles into corresponding values for the iCub and HRP-4C robots. We scaled relative Cartesian positions and used an identity map between the rotational motion of the human and the robot. We also retargeted the center of mass trajectories from human to robot by recurring to normalized offsets that allowed us to reconstruct the robot center of mass position on the ground. Finally we proposed a dynamical correction of the center of mass: if the user performs any motion that can unbalance the robot, their motion is corrected before being transferred to the robot so to guarantee the robot's balance. From the different setups we tested, it seems that tracking the whole-body posture of the user, together with the pose of the end-effectors and the center of mass ground position, produces the most human-like robot behavior. This is possible if dynamic constraints like friction cone constraints are handled by the controller. For very dynamical motions, also a dynamical correction of the retargeted references could be required. We proposed a simple LIP-based center of mass correction for double support motions. Extended simplified models should be used for more complex motions involving the change of contacts.

In Chapter 3, we reviewed the whole-body controllers that have been used in the literature to teleoperate humanoid robots. A common choice is to formulate the control problem as a QP program, which can be used to compute the commands for both position- and torque-controlled robots. In the former case the QP problem represents an inverse dynamics or inverse kinematics

problem, while in the latter it represents a momentum-based control and inverse dynamics problem. We used an IK-based QP controller for the iCub robot and a TSID-based QP for a simulated HRP-4C robot. Each controller is characterized by several parameters that dictate how the reference trajectories of the tasks are tracked, and establish the priority assigned to each task. Tuning the control parameters normally requires a time-consuming trial-and-error procedure that does not necessarily produce effective robot behaviors. For this reason, we focused on automatically optimizing the parameters of the robot controller in simulation to find robust solutions working on the real robot. We proposed the use of stochastic multi-objective optimization algorithms to find solutions that work on the real robot in few trials. We optimized the IK-based QP controller we were using for teleoperating the iCub robot while performing double support motions, by finding a trade-off between maximum tracking performance and minimum ZMP displacement from the center of the support polygon. What robustness criteria to consider for more dynamical movements like walking or running, still remains an open question and should be further investigated.

In Chapter 4, we enhanced the teleoperation system by integrating a higher-level mode for controlling the robot during locomotion. With this integration, the user was able to switch between a low-level teleoperation setting in which they can generate whole-body movements for the robot by means of a motion capture suit and a high-level one in which they can use a joystick to send reference commands to the robot, such as direction and velocity of motion, without dealing with their actual execution. In both cases, the human operator received visual feedback through a virtual reality headset connected to the cameras of the robot. In the high-level teleoperation mode we used an MPC gait generator based on a LIP model. For 3D locomotion more complex models should be employed, and according their nature, a non-linear formulation of the MPC could be required.

Finally in Chapter 5, we addressed one of the problems that emerge when teleoperating robots in non-ideal communication networks. Delays in the transmission of the commands from user to robot and in the reception of the feedback from robot to user, can irremediably disturb the operator. Hence, we proposed a delay compensation approach based on predictive machine learning models that enables to synchronize the user movements with the visual feedback from the robot. The robot anticipates the human motion so that the delayed "anticipated" feedback is synchronized with the operator's commands. In our implementation, we used the ProMPs to represent human motion primitives and also to predict the operator's intentions. We showed that the robot follows the specific user's execution of motion, by continuously updating the prediction of the current motion thanks to the conditioning operator of the ProMPs. In the experiments, we were able to reach objects in ways that had not been demonstrated before (but included in the distribution of the training demonstrations). This feature is extremely valuable because it allows the operator to adapt the commands on the fly to situations different from those in the training. In this way, we preserved the main value of robot teleoperation that makes it possible to exploit the cognition skills of the operator without relying on complex robot perception algorithms, which cannot produce efficient robot behaviors in complex situations. More investigation has to be conducted over the quantity and quality of the data necessary to train the system to achieve this valuable feature in different scenarios.

6.2 Perspective

The system we developed allowed us to perform simple tasks like picking up boxes, reaching objects, opening doors, and imitating the human in the free-space. What prevented us to address more complex teleoperation scenarios, involving more interaction with the environment, was mainly the fragility of the iCub's hands. Despite having many degrees of freedom, they can-

not lift objects heavier than 400g and cannot be used to get a hold on environmental elements like doors or handrails. Differently, the simulated robot HRP-4C exhibits limited dexterity in the arms due to the limited degrees of freedom of the hands and of the wrists (only two each). More complex and robust hands however, would require more information of the state of the robot and its environment.

Haptic feedback can provide extremely valuable information during manipulation. While safe physical interaction with the environment may be ensured by the robot low-level control without haptic feedback, this feedback information is critical to enhance the remote control capabilities of the operator [20, 154]. Typically, haptic feedback is localized in the end-effectors, where most of the interaction occurs; in the case of teleoperated humanoids, whole-body haptic feedback should be considered [149], possibly by means of wearable vibro-tactile devices [32].

In order to perform more dynamical motions, forces at the contacts should be controlled, i.e., one should recur to torque control. With respect to position control, this provides a number of benefits, especially for humanoid robots. Torque control provides improved tracking performances and lower feedback gains can be used with the controller, resulting in higher compliance of the system. This higher compliance enables safer human-robot interaction and to automatically adapt to uncertain environment (e.g. walking on uneven terrain). However, achieving reliable robot behaviors with torque control still represents a challenge for the majority of the existing robotic platforms. In fact, while in simulation the joint torque control can be almost perfect, on the real robot there are errors in the actuation chain model, sensor noise, limited torque bandwidth and/or delays [153]. Hence many of the behaviors obtained in simulation cannot be transferred easily on the real robot.

Another future improvement of the system we proposed in this thesis, could be related to the use of shared-control approaches [148, 166]. These make it possible to combine the capabilities of the robot and of the operator to achieve complex tasks more efficiently. The operator guides the robot with their high-level decisions and advanced cognition skills, while the robot performs autonomously or semi-autonomously the tasks exploiting its precision skills which can often outperform those retargeted from the human operator. For example in [148], the robot modifies the orientation of its end-effectors to help the operator accomplish bi-manual tasks more efficiently. An interesting alternative that has been recently proposed is to guide the operator's execution of a task through haptic feedback [16, 114, 146]. In this case, the haptic feedback no longer represents the actual forces experienced at the robot side but it is used to guide the human motion. For example, the level of robotic guidance can be modulated based on the prediction of the human motion intentions [114] or based on the distance to the target object [16].

In all these approaches, the teleoperation becomes an instance of human-robot collaboration. This collaboration is not one-sided (i.e., the operator controls the robot that passively follows the user commands), but both parts help each other to accomplish a task more fluently and efficiently. In this sense also our delay compensation approach can be identified as a collaboration strategy. As long as the human operator is the one taking the high level decisions for the robot and is not deprived of the possibility of taking full control of it, any assistance from the robot can only be beneficial. In fact, we assume that if the operator would have to put too much effort in controlling the robot when accomplishing easy tasks, then the embodiment experience would immediately break, making the operator frustrated or lose their patience. This will be a key feature in future telexistence applications.

The work of this thesis was funded by the European project AnDy [9] [82], which had the goal to endow robots with the ability to control physical collaboration through intentional interaction

[9] Advancing anticipatory behaviors in Dynamic human-robot collaboration.

and anticipatory behaviors. The teleoperation system we presented in the thesis could be used to generate human-like collaborative policies for the robot in future collaboration scenarios. Furthermore, the prediction-based delay compensation approach we proposed that allows the robot to anticipate the human motion based on predictive machine learning models, could be applied on exoskeletons and cobots in the industry, increasing the productivity, flexibility and safety of the human workers. The predictive models in fact, could be used not only to compensate for the delays but also to anticipate human behaviors in physical collaboration scenarios, to prevent fatal injuries, to limit the effort of the human workers and to improve the ergonomics of their movements.

6.3 Future Challenges

The next great challenge in humanoid teleoperation is the extension to more complex and realistic scenarios, closer to those encountered in the real world, such as performing loco-manipulation on an uneven terrain in a search and rescue or in a bomb disposal operation. Clearly, in these more realistic situations, controlling simultaneously the upper-body and the lower part of the robot is crucial. Extending the whole-body retargeting also to these kind of situations, would be ideal. However, it is not intuitive since they involve locomotion that is not limited to walking on flat ground. The main problem associated with retargeting locomotions at a whole-body level is related to the setup at the user side. The user cannot walk around, because of the limited workspace and because the environment at the user side is different than that at the robot side. Walking platforms have been proposed in the literature [43], but they make sense only for walking on flat ground and constrain the upper-body motion of the user. Another proposed solution has been to use whole-body exoskeletons [80]. Also in this case, the operator wearing the exoskeleton cannot walk around because of the limited workspace. The solution in [80] of fixing the floating base of the exoskeleton, seems to generate unnatural user motions. The dynamics information from the user motion loses its value and cannot be used to control the robot efficiently. An ideal solution would be to have a *metamorphic walking platform* that gradually shifts the user at its center and rearranges its structure according to the robot environment (next contact locations). If for example, there is a step in front of the robot, the walking platform recreates that step in front of the human. By combining this walking platform to a balance feedback interface like that proposed in [151], could also enable the retarget of motions that involve contacts with the upper-body, such as leaning against a wall, while providing some force to the user to prevent them from falling.

An alternative and more simple solution, could be to still rely on existing high-level interfaces like joysticks to guide periodic locomotions like walking and climbing stairs, and then use whole-body retargeting for the other tasks with limited locomotion. A promising solution, for the latter would be to control the robot focusing solely on the manipulation, while letting the robot autonomously reposition its feet in order to facilitate the manipulation and to take into account the different environment at user and robot site.

Bibliography

[1] ASIMO by Honda. The World's Most Advanced Humanoid Robot. https://asimo.honda.com/.

[2] Atlas by Boston Dynamics. https://www.bostondynamics.com/atlas.

[3] Canadarm. Canadian Space Agency. https://www.asc-csa.gc.ca/eng/canadarm/default.asp.

[4] Census of fatal occupational injuries summary, 2017. https://www.bls.gov/news.release/cfoi.nr0.htm.

[5] Ergonomics of human-system interaction — Part 11: Usability: definitions and concepts. https://www.iso.org/.

[6] Iproute2. The Linux Foundation. https://wiki.linuxfoundation.org/networking/iproute2.

[7] Kinovea. A microscope for your videos. https://www.kinovea.org/.

[8] Mc_rtc. A framework for multi-contact real-time control. https://jrl-umi3218.github.io/mc_rtc/index.html.

[9] National Aeronautics and Space Administration (NASA)-Centennial Challenges Program-Space Robotics Challenge Phase 2. https://www.fbo.gov/index?s=opportunity&mode=form&id=6b4c9a474d2edd9934be7008edc665cd&tab=core&_cview=0.

[10] Occupational safety and health administration. more rescuers die than the original victims. https://www.osha.gov/redirect?p_table=PREAMBLES&p_id=839.

[11] Russian humanoid robot to pilot soyuz capsule to iss this week. https://spectrum.ieee.org/automaton/robotics/space-robots/russian-humanoid-robot-to-pilot-soyuz-capsule-to-iss-this-week.

[12] Teleoperation of the humanoid robot, space teleoperation challenges and approaches in GITAI. https://youtu.be/XXz8CdkXPiI.

[13] Virtual Desktop. Your PC in VR. https://www.vrdesktop.net/.

[14] WALK-MAN: Whole-body Adaptive Locomotion and Manipulation, EU Project. https://www.walk-man.eu/.

[15] Workshop on teleoperation of humanoid robots. `https://ami-iit.github.io/WS_teleoepration_humanoids/`.

[16] Firas Abi-Farraj, Claudio Pacchierotti, Oleg Arenz, Gerhard Neumann, and Paolo Robuffo Giordano. A haptic shared-control architecture for guided multi-target robotic grasping. *IEEE Transactions on Haptics*, 13(2):270–285, 2020.

[17] F. Abi-Farrajl, B. Henze, A. Werner, M. Panzirsch, C. Ott, and M. A. Roa. Humanoid teleoperation using task-relevant haptic feedback. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Oct 2018.

[18] Ahmed Aboudonia, Nicola Scianca, Daniele De Simone, Leonardo Lanari, and Giuseppe Oriolo. Humanoid gait generation for walk-to locomotion using single-stage MPC. In *17th IEEE-RAS International Conference on Humanoid Robots*, pages 178–183, 2017.

[19] Juul Achten and Asker E Jeukendrup. Heart rate monitoring. *Sports medicine*, 33(7):517–538, 2003.

[20] Marco Aggravi, Ahmed Alaaeldin Said Elsherif, Paolo Robuffo Giordano, and Claudio Pacchierotti. Haptic-enabled decentralized control of a heterogeneous human-robot team for search and rescue in partially-known environments. *IEEE Robotics and Automation Letters*, 6(3):4843–4850, March 2021.

[21] Arash Ajoudani, Nikos Tsagarakis, and Antonio Bicchi. Tele-impedance: Teleoperation with impedance regulation using a body–machine interface. *The Int. J. of Robotics Research*, 31(13):1642–1656, 2012.

[22] Baris Akgun, Kaushik Subramanian, and Andrea Lockerd Thomaz. Novel interaction strategies for learning from teleoperation. In *AAAI Fall Symposium: Robots Learning Interactively from Human Teachers*, 2012.

[23] Adel Alharbi, Ayoub Bahnasse, and Mohamed Talea. A Comparison of VoIP Performance Evaluation on different environments Over VPN Multipoint Network. *International Journal of Computer Science and Network Security*, 17:123–128, 05 2017.

[24] R. Antonova, S. Cruciani, C. Smith, and D. Kragic. Reinforcement learning for pivoting task. *CoRR*, abs/1703.00472, 2017.

[25] Mohammad Anvaripour, Mahta Khoshnam, Carlo Menon, and Mehrdad Saif. *Frontiers in Robotics and AI*, 7:183–183, 2020.

[26] Christopher Atkeson, Benzun Pious Wisely Babu, Nandan Banerjee, Dmitry Berenson, Christoper Bove, Xiongyi Cui, Mathew Dedonato, Ruixiang Du, Siyuan Feng, Perry Franklin, M. Gennert, Joshua Graff, Peng He, Aaron Jaeger, Joohyung Kim, Kevin Knoedler, Lening Li, Chenggang Liu, Xianchao Long, and X. Xinjilefu. *What Happened at the DARPA Robotics Challenge Finals*, pages 667–684. Springer Tracts in Advanced Robotics, 2018.

[27] Ko Ayusawa and Eiichi Yoshida. Motion retargeting for humanoid robots based on simultaneous morphing parameter identification and motion optimization. *IEEE Trans. on Robotics*, 33(6):1343–1357, 2017.

[28] A. K. Bejczy, W. S. Kim, and S. C. Venema. The phantom robot: predictive displays for teleoperation with time delay. *in Proceedings of the IEEE International Conference on Robotics and Automation*, pages 546–551 vol.1, 1990.

[29] Alberto Bemporad. Predictive control of teleoperated constrained systems with unbounded communication delays. In *IEEE Conference on Decision and Control*, Dec 1998.

[30] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.

[31] A. Brygo, I. Sarakoglou, N. Tsagarakis, and D. G. Caldwell. Tele-manipulation with a humanoid robot under autonomous joint impedance regulation and vibrotactile balancing feedback. In *IEEE/RAS Int. Conf. on Humanoid Robots*, Nov 2014.

[32] Anais Brygo, Ioannis Sarakoglou, Nadia Garcia-Hernandez, and Nikolaos Tsagarakis. Humanoid robot teleoperation with vibrotactile based balancing feedback. In *Int. Conf. on Human Haptic Sensing and Touch Enabled Computer Applications*, pages 266–275.

[33] Stephane Caron, Abderrahmane Kheddar, and Olivier Tempier. Stair climbing stabilization of the hrp-4 humanoid robot using whole-body admittance control. In *IEEE Int. Conf. on Robotics and Automation*, pages 277–283, 05 2019.

[34] Justin Carpentier et al. Multi-contact Locomotion of Legged Robots in Complex Environments – The Loco3D project. In *RSS Workshop on Challenges in Dynamic Legged Locomotion*, page 3p., Boston, July 2017.

[35] Marie Charbonneau, Valerio Modugno, Francesco Nori, Giuseppe Oriolo, Daniele Pucci, and Serena Ivaldi. Learning robust task priorities of QP-based whole-body torque-controllers. In *IEEE-RAS Int. Conf. on Humanoid Robots*, 2018.

[36] Y. Choi, D. Kim, Y. Oh, and B. You. Posture/walking control for humanoid robot based on kinematic resolution of com jacobian with embedded motion. *IEEE Transactions on Robotics*, 23(6):1285–1293, 2007.

[37] Rubana H Chowdhury, Mamun BI Reaz, Mohd Alauddin Bin Mohd Ali, Ashrif AA Bakar, Kalaivani Chellappan, and Tae G Chang. Surface electromyography signal processing and classification techniques. *Sensors*, 13(9), 2013.

[38] R. Cisneros, M. Morisawa, S. Nakaoka, K. Kaneko, S. Kajita, T. Sakaguchi, and F. Kanehiro. Enabling a teleoperated humanoid robot to pass through debris-filled terrain using manipulation. In *IEEE/RAS Int. Conf. on Humanoid Robots*, Nov 2016.

[39] Mark Claypool and Jonathan Tanner. The effects of jitter on the peceptual quality of video. *in Proceedings of the seventh ACM international conference on Multimedia (Part 2)*, pages 115–118, 1999.

[40] Debora Clever et al. COCoMoPL: A novel approach for humanoid walking generation combining optimal control, movement primitives and learning and its transfer to the real robot HRP-2. *IEEE Robotics and Automation Letters*, 2(2):977–984, 2017.

[41] Enric Corona, Albert Pumarola, Guillem Alenya, and Francesc Moreno-Noguer. Context-aware human motion prediction. In *in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6992–7001, 2020.

[42] K. Darvish, **L. Penco**, J. Ramos, R. Cisneros, J. Pratt, E. Yoshida, S. Ivaldi, and D. Pucci. Teleoperation of humanoid robots: A survey. 2022.

[43] Kourosh Darvish, Yeshasvi Tirupachuri, Giulio Romualdi, Lorenzo Rapetti, Diego Ferigo, Francisco Javier Andrade Chavez, and Daniele Pucci. Whole-body geometric retargeting for humanoid robots. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 679–686, 2019.

[44] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[45] Kalyanmoy Deb and Deb Kalyanmoy. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., NY, USA, 2001.

[46] N. Dehio, R. F. Reinhart, and J. J. Steil. Multiple task optimization with a mixture of controllers for motion generation. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 6416–6421, Sept 2015.

[47] Niels Dehio and Jochen Steil. Dynamically-consistent generalized hierarchical control. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1141–1147, May 2019.

[48] Andrea Del Prete and Nicolas Mansard. Robustness to Joint-Torque Tracking Errors in Task-Space Inverse Dynamics. *IEEE Transaction on Robotics*, 32(5):1091 – 1105, 2016.

[49] Andrea Del Prete, Francesco Nori, Giorgio Metta, and Lorenzo Natale. Prioritized motion–force control of constrained fully-actuated robots: "task space inverse dynamics". *Robotics and Autonomous Systems*, 63:150–157, 2015.

[50] Oriane Dermy, Maxime Chaveroche, Francis Colas, François Charpillet, and Serena Ivaldi. Prediction of human whole-body movements with ae-promps. *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 572–579, 2018.

[51] Oriane Dermy, Alexandros Paraschos, Marco Ewerton, Jan Peters, François Charpillet, and Serena Ivaldi. Prediction of intention during interaction with iCub with probabilistic movement primitives. *Frontiers in Robotics and AI*, 4:45–45, 2017.

[52] Myron A Diftler, JS Mehling, Muhammad E Abdallah, Nicolaus A Radford, Lyndon B Bridgwater, Adam M Sanders, Roger Scott Askew, D Marty Linn, John D Yamokoski, FA Permenter, et al. Robonaut 2-the first humanoid robot in space. In *2011 IEEE international conference on robotics and automation*, pages 2178–2183. IEEE, 2011.

[53] Anca D. Dragan, Kenton C.T. Lee, and Siddhartha S. Srinivasa. Legibility and predictability of robot motion. In *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 301–308, 2013.

[54] Anca D. Dragan and Siddhartha S. Srinivasa. Formalizing assistive teleoperation. In *Robotics: Science and Systems*, 2012.

[55] Mohamed Elobaid, Yue Hu, Giulio Romualdi, Stefano Dafarra, Jan Babic, and Daniele Pucci. Telexistence and teleoperation for walking humanoid robots. *IntelliSys*, 1038:1106–1121, 2019.

[56] J. Englsberger, C. Ott, and A. Albu-Schäffer. Three-dimensional bipedal walking control based on divergent component of motion. *IEEE Transactions on Robotics*, 31(2):355–368, 2015.

[57] K. Erbatur, A. Okazaki, K. Obiya, T. Takahashi, and A. Kawamura. A study on the zero moment point measurement for biped walking robots. In *7th International Workshop on Advanced Motion Control. Proceedings (Cat. No.02TH8623)*, pages 431–436, 2002.

[58] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *IJRR*, 33(7):1006–1028, 2014.

[59] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *The International Journal of Robotics Research*, 33(7):1006–1028, 2014.

[60] P. Evrard, N. Mansard, O. Stasse, A. Kheddar, T. Schauß, C. Weber, A. Peer, and M. Buss. Intercontinental, multimodal, wide-range tele-cooperation using a humanoid robot. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Oct 2009.

[61] A. Di Fava, K. Bouyarmane, K. Chappellet, E. Ruffaldi, and A. Kheddar. Multi-contact motion retargeting from human to humanoid robot. In *IEEE/RAS Int. Conf. on Humanoid Robots*, Nov 2016.

[62] A. Di Fava, K. Bouyarmane, K. Chappellet, E. Ruffaldi, and A. Kheddar. Multi-contact motion retargeting from human to humanoid robot. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 1081–1086, Nov 2016.

[63] W. R. Ferrell. Remote manipulation with transmission delay. *IEEE Transactions on Human Factors in Electronics*, HFE-6(1):24–32, 1965.

[64] W. R. Ferrell and T. B. Sheridan. Supervisory control of remote manipulation. *IEEE Spectrum*, 4(10):81–88, 1967.

[65] Michele Garetto and Don Towsley. Modeling, simulation and measurements of queuing delay under long-tail internet traffic. In *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '03, page 47–57, New York, NY, USA, 2003. Association for Computing Machinery.

[66] Oliver Glauser, Shihao Wu, Daniele Panozzo, Otmar Hilliges, and Olga Sorkine-Hornung. Interactive hand pose estimation using a stretch-sensing soft glove. *ACM Trans. on Graphics*, 38(4):1–15, 2019.

[67] R C Goertz. Fundamentals of general-purpose remote manipulators. *Nucleonics (U.S.) Ceased publication*, Vol: 10, No. 11, 1952.

[68] Waldez Gomes, Vishnu Radhakrishnan, **Luigi Penco**, Valerio Modugno, Jean-Baptiste Mouret, and Serena Ivaldi. Humanoid whole-body movement optimization from retargeted human motions. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 178–185, 2019.

[69] Michael A Goodrich and Alan C Schultz. Human-robot interaction: a survey. *Foundations and Trends® in Human–Computer Interaction*, 1(3):203–275, 2007.

[70] O. Gurewitz, I. Cidon, and M. Sidi. One-way delay estimation using network-wide measurements. *IEEE Transactions on Information Theory*, 52(6):2710–2724, 2006.

[71] Andrei Herdt, Holger Diedam, Pierre-Brice Wieber, Dimitar Dimitrov, Katja Mombaur, and Moritz Diehl. Online walking motion generation with automatic footstep placement. *Advanced Robotics*, 24(5-6):719–737, 2010.

[72] Miguel Hernando and Ernesto Gambao. *Advances in Telerobotics*, chapter Teleprograming: Capturing the Intention of the Human Operator, pages 303–320. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[73] E. M. Hoffman, A. Laurenzi, L. Muratore, N. G. Tsagarakis, and D. G. Caldwell. Multi-priority cartesian impedance control based on quadratic programming optimization. In *IEEE International Conference on Robotics and Automation*, pages 309–315, 2018.

[74] Enrico Mingo Hoffman, Brice Clement, Chengxu Zhou, Nikos G Tsagarakis, Jean-Baptiste Mouret, and Serena Ivaldi. Whole-Body Compliant Control of iCub: first results with OpenSoT. In *IEEE International Conference on Robotics and Automation 2018 - Workshop*, 2018.

[75] Enrico Mingo Hoffman, Alessio Rocchi, Arturo Laurenzi, and Nikos G Tsagarakis. Robot control for dummies: Insights and examples using OpenSoT. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 736–741, 2017.

[76] Toke Høiland-Jørgensen, Bengt Ahlgren, Per Hurtig, and Anna Brunstrom. Measuring latency variation in the internet. *in Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies*, pages 473–480, 2016.

[77] Kai Hu, Christian Ott, and Dongheui Lee. Online human walking imitation in task and joint space based on quadratic programming. In *IEEE Int. Conf. on Robotics and Automation*, pages 3458–3464, 2014.

[78] Päivi Hämäläinen, Jukka Takala, and Kaija Saarela. Global estimates of occupational accidents. *Safety Science*, 44:137–156, 02 2006.

[79] Y. Ishiguro, K. Kojima, F. Sugai, S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba. High speed whole body dynamic motion experiment with real time master-slave humanoid robot system. In *IEEE Int. Conf. on Robotics and Automation*, May 2018.

[80] Yasuhiro Ishiguro, Tasuku Makabe, Yuya Nagamatsu, Yuta Kojio, Kunio Kojima, Fumihito Sugai, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. Bilateral humanoid teleoperation system using whole-body exoskeleton cockpit TABLIS. *IEEE Robotics and Automation Letters*, 5(4):6419–6426, 2020.

[81] Serena Ivaldi, Jan Babič, Michael Mistry, and Robin Murphy. Special issue on whole-body control of contacts and dynamics for humanoid robots. *Autonomous Robots*, 40(3):425–428, 2016.

[82] Serena Ivaldi, Lars Fritzsche, Jan Babič, Freek Stulp, Michael Damsgaard, Bernhard Graimann, Giovanni Bellusci, and Francesco Nori. Anticipatory models of human movements and dynamics: the roadmap of the AnDy project. In *DHM*, 2017.

[83] Serena Ivaldi, Matteo Fumagalli, Marco Randazzo, Francesco Nori, Giorgio Metta, and Giulio Sandini. Computing robot internal/external wrenches by means of inertial, tactile and f/t sensors: Theory and implementation on the icub. In *Humanoids*, pages 521–528, 2011.

[84] N. Jaffar, A.H. Abdul-Tharim, I.F. Mohd-Kamar, and N.S. Lop. A literature review of ergonomics risk factors in construction industry. *Procedia Engineering*, 20:89–97, 2011. 2nd International Building Control Conference.

[85] Matthew Johnson, Brandon Shrewsbury, Sylvain Bertrand, Duncan Calvert, Tingfan Wu, Daniel Duran, Douglas Stephen, Nathan Mertins, John Carff, William Rifenburgh, et al. Team IHMC's lessons learned from the DARPA Robotics Challenge: Finding data in the rubble. *J. of Field Robotics*, 34(2):241–261, 2017.

[86] Matthew Johnson, Brandon Shrewsbury, Sylvain Bertrand, Duncan Calvert, Tingfan Wu, Daniel Duran, Douglas Stephen, Nathan Mertins, John Carff, William Rifenburgh, Jesper Smith, Chris Schmidt-Wetekam, Davide Faconti, Alex Graber-Tilton, Nicolas Eyssette, Tobias Meier, Igor Kalkov, Travis Craig, Nick Payton, Stephen McCrory, Georg Wiedebach, Brooke Layton, Peter Neuhaus, and Jerry Pratt. *Team IHMC's Lessons Learned from the DARPA Robotics Challenge: Finding Data in the Rubble*, pages 71–102. Springer International Publishing, 2018.

[87] Steven Jens Jorgensen, Michael W. Lanighan, Sylvain S. Bertrand, Andrew Watson, Joseph S. Altemus, R. Scott Askew, Lyndon Bridgwater, Beau Domingue, Charlie Kendrick, Jason Lee, and et al. Deploying the NASA Valkyrie humanoid for IED response: An initial approach and evaluation summary. In *IEEE/RAS Int. Conf. on Humanoid Robots*, page 1–8, Oct 2019.

[88] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *IEEE International Conference on Robotics and Automation*, pages 1620–1626, 2003.

[89] S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, and K. Yokoi. Biped walking stabilization based on linear inverted pendulum tracking. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4489–4496, 2010.

[90] Shuuji Kajita, Hirohisa Hirukawa, Kensuke Harada, and Kazuhito Yokoi. *Introduction to Humanoid Robotics*. Springer Publishing Company Inc., 2014.

[91] Shuuji Kajita, Kenji Kaneko, Fumio Kanehiro, Kensuke Harada, Mitsuharu Morisawa, Shin'ichiro Nakaoka, Kanako Miura, Kiyoshi Fujiwara, Ee Sian Neo, and Isao Hara. Cybernetic human HRP-4C: A humanoid robot with human-like proportions. In *ISRR*, 2009.

[92] Eric R. Kandel, Jams H. Schwartz, Thomas M. Jessell, Steven A. Siegelbaum, and A. J. Hudspeth. *Principles of Neural Science*. Mc Graw Hill Medical, fifth edition, 2013.

[93] Peter Kazanzides, Balazs P. Vagvolgyi, Will Pryor, Anton Deguet, Simon Leonard, and Louis L. Whitcomb. Teleoperation and visualization interfaces for remote intervention in space. *Frontiers in Robotics and AI*, 8, 2021.

[94] Arvid QL Keemink, Herman van der Kooij, and Arno HA Stienen. Admittance control for physical human–robot interaction. *The Int. J. of Robotics Research*, 37(11):1421–1444, 2018.

[95] F. Keith, P. Wieber, N. Mansard, and A. Kheddar. Analysis of the discontinuities in prioritized tasks-space control under discreet task scheduling operations. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3887–3892, Sep. 2011.

[96] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, February 1987.

[97] Doik Kim, Bum-Jae You, and Sang-Rok Oh. *Whole Body Motion Control Framework for Arbitrarily and Simultaneously Assigned Upper-Body Tasks and Walking Motion*, pages 87–98. Springer, 2013.

[98] Sanghyun Kim, Keunwoo Jang, Suhan Park, Yisoo Lee, Sang Yup Lee, and Jaeheung Park. Continuous task transition approach for robot controller based on hierarchical quadratic programming. *IEEE Robotics and Automation Letters*, 2019.

[99] Taewoo Kim and Joo-Haeng Lee. C-3PO: Cyclic-Three-Phase optimization for human-robot motion retargeting based on reinforcement learning. In *IEEE Int. Conf. on Robotics and Automation*, May-Aug 2020.

[100] Twan Koolen, Sylvain Bertrand, Gray Thomas, Tomas De Boer, Tingfan Wu, Jesper Smith, Johannes Englsberger, and Jerry Pratt. Design of a momentum-based control framework and application to the humanoid robot atlas. *International Journal of Humanoid Robotics*, 13:1650007–1, 03 2016.

[101] P. Kremer, T. Wimbock, J. Artigas, S. Schatzle, K. Johl, F. Schmidt, C. Preusche, and G. Hirzinger. Multimodal telepresent control of DLR's Rollin' JUSTIN. In *IEEE Int. Conf. on Robotics and Automation*, May 2009.

[102] Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 40(3):429–455, Mar 2016.

[103] Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Auton. Robots*, 40(3):429–455, 2016.

[104] Jeongseok Lee et al. Dart: Dynamic animation and robotics toolkit. *J. Open Source Software*, 3:500, 2018.

[105] Zhi Li, Peter Moran, Qingyuan Dong, Ryan J Shaw, and Kris Hauser. Development of a tele-nursing mobile manipulator for remote care-giving in quarantine areas. In *IEEE Int. Conf. on Robotics and Automation*, pages 3581–3586, May-Jun 2017.

[106] Minas V Liarokapis, PK Artemiadis, CP Bechlioulis, and KJ Kyriakopoulos. Directions, methods and metrics for mapping human to robot motion with functional anthropomorphism: A review. *School of Mechanical Engineering, National Technical University of Athens, Tech. Rep*, 2013.

[107] Neal Lii, Cornelia Riecke, Daniel Leidner, Simon Schätzle, Peter Schmaus, Bernhard Weber, Thomas Krueger, Martin Stelzer, Armin Wedler, and Gerhard Grunwald. The robot as an avatar or co-worker? an investigation of the different teleoperation modalities through the KONTUR-2 and METERON SUPVIS Justin space telerobotic missions. In *Int. Astronautical Cong.*, Oct 2018.

[108] Neal Y. Lii, Daniel Leidner, André Schiele, Peter Birkenkampf, Benedikt Pleintinger, and Ralph Bayer. Command robots from orbit with supervised autonomy: An introduction to the meteron supvis-justin experiment. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*, HRI'15 Extended Abstracts, page 53–54, New York, NY, USA, 2015. Association for Computing Machinery.

[109] Neal Y. Lii, Daniel Sebastian Leidner, André Schiele, Peter Birkenkampf, Ralph Bayer, Benedikt Pleintinger, Andreas Meissner, and Andreas Balzer. Simulating an extraterrestrial environment for robotic space exploration: The METERON SUPVIS-JUSTIN telerobotic experiment and the SOLEX proving ground. 2015.

[110] Bryan Lim and Stefan Zohren. Time series forecasting with deep learning: A survey. *Philosophical Transactions of the Royal Society A.*, 379:20200209–20200209, 2021.

[111] Yiping Liu, Patrick M. Wensing, David E. Orin, and Yuan F. Zheng. Dynamic walking in a humanoid robot based on a 3d actuated dual-slip model. In *IEEE International Conference on Robotics and Automation*, pages 5710–5717, 2015.

[112] R. Lober, V. Padois, and O. Sigaud. Variance modulated task prioritization in whole-body control. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3944–3949, Sep. 2015.

[113] Dylan P. Losey, Craig G. McDonald, Edoardo Battaglia, and Marcia K. O'Malley. A Review of Intent Detection, Arbitration, and Communication Aspects of Shared Control for Physical Human–Robot Interaction. *Applied Mechanics Reviews*, 70(1):10804–10804, 02 2018.

[114] Kim Tien Ly, Mithun Poozhiyil, Harit Pandya, Gerhard Neumann, and Ayse Kucukyilmaz. Intent-aware predictive haptic guidance and its application to shared control teleoperation. In *2021 30th IEEE International Conference on Robot Human Interactive Communication (RO-MAN)*, pages 565–572, 2021.

[115] Y. Tan M. Liu and V. Padois. Generalized hierarchical control. In *Autonomous Robots*, pages vol. 40, pp. 17–31, 2016.

[116] Karl F. MacDorman and Hiroshi Ishiguro. The uncanny advantage of using androids in cognitive and social science research. *Interaction Studies*, 7(3):297–337, 2006.

[117] Guilherme J Maeda, Gerhard Neumann, Marco Ewerton, Rudolf Lioutikov, Oliver Kroemer, and Jan Peters. Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks. *Autonomous Robots*, 41(3):593–612, 2017.

[118] Nicolas Mansard, Olivier Stasse, Paul Evrard, and Abderrahmane Kheddar. A versatile generalized inverted kinematics implementation for collaborative working humanoid robots: The stack of tasks. In *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pages 1–6. IEEE, 2009.

[119] S. Mason, L. Righetti, and S. Schaal. Full dynamics lqr control of a humanoid robot: An experimental study on balancing and squatting. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 374–379, 2014.

[120] S. Mason, N. Rotella, S. Schaal, and L. Righetti. Balancing and walking using full dynamics lqr control with contact constraints. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 63–68, 2016.

[121] D. Mills, J. Martin, J. Burbank, and W. Kasch. Network time protocol version 4: Protocol and algorithms specification. RFC 5905, RFC Editor, June 2010. `http://www.rfc-editor.org/rfc/rfc5905.txt`.

[122] P. Mitra and G. Niemeyer. Mediating time delayed teleoperation with user suggested models: Implications and comparative study. *2008 Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 343–350, 2008.

[123] V. Modugno, U. Chervet, G. Oriolo, and S. Ivaldi. Learning soft task priorities for safe control of humanoid robots with constrained stochastic optimization. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 101–108, 2016.

[124] V. Modugno, G. Neumann, E. Rueckert, G. Oriolo, J. Peters, and S. Ivaldi. Learning soft task priorities for control of redundant robots. In *ICRA*, pages 221–226, May 2016.

[125] J.-B. Mouret and S. Doncieux. Sferesv2: Evolvin' in the multi-core world. In *IEEE Cong. Evolutionary Computation*, pages 1–8, July 2010.

[126] Lorenzo Natale, Chiara Bartolozzi, Daniele Pucci, Agnieszka Wykowska, and Giorgio Metta. iCub: The not-yet-finished story of building a robot child. *Science Robotics*, 2(13):eaaq1026, 2017.

[127] Ernst Niedermeyer and FH Lopes da Silva. *Electroencephalography: basic principles, clinical applications, and related fields*. Lippincott Williams & Wilkins, 2005.

[128] OECD. *Occupational accidents in oecd countries, OECD Employment Outlook 1989, chapter 4*. 1989.

[129] Toshio Ohhashi, Masao Sakaguchi, and Takao Tsuda. Human perspiration measurement. *Physiological measurement*, 19(4):449, 1998.

[130] Boris Oklander and M. Sidi. Jitter buffer analysis. In *in Proceedings of the International Conference on Computer Communications and Networks, ICCCN*, pages 1 – 6, 09 2008.

[131] K. Otani and K. Bouyarmane. Adaptive whole-body manipulation in human-to-humanoid multi-contact motion retargeting. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 446–453, Nov 2017.

[132] Kazuya Otani, Karim Bouyarmane, and Serena Ivaldi. Generating assistive humanoid motions for co-manipulation tasks with a multi-robot quadratic program controller. In *IEEE International Conference on Robotics and Automation*, 2018.

[133] C. Ott, M. A. Roa, and G. Hirzinger. Posture and balance control for biped robots based on contact force optimization. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 26–33, 2011.

[134] Christian Ott, Alexander Dietrich, and Alin Albu-Schäeffer. An overview of null space projections for redundant, torque-controlled robots. *Automatica*, 53:416–423, 03 2015.

[135] Jörg Ott, Stephan Wenger, Noriyuki Sato, Carsten Burmeister, and Jose Rey. Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF). *RFC*, 4585:1–51, 2006.

[136] A. Paraschos, R. Lioutikov, J. Peters, and G. Neumann. Probabilistic prioritization of movement primitives. *IEEE Robotics and Automation Letters*, 2(4):2294–2301, Oct 2017.

[137] Alexandros Paraschos, Christian Daniel, Jan Peters, and Gerhard Neumann. Using probabilistic movement primitives in robotics. *Autonomous Robots*, 42:529–551, 07 2018.

[138] Antonio Rafael Sabino Parmezan, Vinicius MA Souza, and Gustavo EAPA Batista. Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. *Information sciences*, 484:302–337, 2019.

[139] Patrice Pavis. *Dictionary of the theatre: Terms, concepts, and analysis*. University of Toronto Press, 1998.

[140] A. Peer, S. Hirche, C. Weber, I. Krause, M. Buss, S. Miossec, P. Evrard, O. Stasse, E. S. Neo, A. Kheddar, and K. Yokoi. Intercontinental multimodal tele-cooperation using a humanoid robot. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 405–411, Sept 2008.

[141] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph.*, 37(4), jul 2018.

[142] L.F. Peñin, K. Matsumoto, and Kōkū Uchū Gijutsu Kenkyūjo. *Teleoperation with Time Delay: A Survey and Its Use in Space Robotics*. National Aerospace Laboratory, 2002.

[143] Luka Peternel and Jan Babič. Learning of compliant human–robot interaction using full-body haptic interface. *Advanced Robotics*, 27, 09 2013.

[144] R. M. Pierce and K. J. Kuchenbecker. A data-driven method for determining natural human-robot motion mappings in teleoperation. In *IEEE/RAS Int. Conf. on Biomedical Robotics and Biomechatronics*, pages 169–176, Jun 2012.

[145] J. Pratt, J. Carff, S. Drakunov, and A. Goswami. Capture point: A step toward humanoid push recovery. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 200–207, Dec 2006.

[146] Rahaf Rahal, Firas Abi-Farraj, Paolo Robuffo Giordano, and Claudio Pacchierotti. Haptic shared-control methods for robotic cutting under nonholonomic constraints. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8151–8157, 2019.

[147] Rahaf Rahal, Giulia Matarese, Marco Gabiccini, Alessio Artoni, Domenico Prattichizzo, Paolo Robuffo Giordano, and Claudio Pacchierotti. Caring about the human operator: Haptic shared control for enhanced user comfort in robotic telemanipulation. *IEEE Transactions on Haptics*, 13(1):197–203, 2020.

[148] Daniel Rakita, Bilge Mutlu, Michael Gleicher, and Laura M. Hiatt. Shared control–based bimanual robot manipulation. *Science Robotics*, 4(30), 2019.

[149] J. Ramos and S. Kim. Humanoid dynamic synchronization through whole-body bilateral feedback teleoperation. *IEEE Trans. on Robotics*, 34(4):953–965, 2018.

[150] J. Ramos, A. Wang, W. Ubellacker, J. Mayo, and S. Kim. A balance feedback interface for whole-body teleoperation of a humanoid robot and implementation in the HERMES system. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 844–850, Nov 2015.

[151] Joao Ramos and Sangbae Kim. Dynamic locomotion synchronization of bipedal robot and human operator via bilateral feedback teleoperation. *Science Robotics*, 4(35), 2019.

[152] O. E. Ramos, N. Mansard, O. Stasse, C. Benazeth, S. Hak, and L. Saab. Dancing humanoid robots: Systematic use of OSID to compute dynamically consistent movements following a motion capture pattern. *IEEE Robotics Automation Mag.*, 22(4), 2015.

[153] N. Ramuzat, G. Buondonno, S. Boria, and O. Stasse. Comparison of position and torque whole-body control schemes on the humanoid robot talos. In *2021 20th International Conference on Advanced Robotics (ICAR)*, pages 785–792, 2021.

[154] Grégoire Richard, Thomas Pietrzak, Ferran Argelaguet, Anatole Lécuyer, and Géry Casiez. Studying the role of haptic feedback on virtual embodiment in a drawing task. *Frontiers in Virtual Reality*, 1, 2021.

[155] A. Rocchi, E. M. Hoffman, D. G. Caldwell, and N. G. Tsagarakis. Opensot: A whole-body control library for the compliant humanoid robot coman. In *IEEE International Conference on Robotics and Automation*, pages 6248–6253, May 2015.

[156] Daniel Roetenberg, Henk Luinge, and Per Slycke. Xsens MVN: Full 6DOF human motion tracking using miniature inertial sensors. *Xsens Motion Technologies BV, Tech. Rep*, 3, 2009.

[157] Daniel Roetenberg, Henk Luinge, and Per Slycke. MVN User Manual. *Xsens Motion Technologies BV, Tech. Rep*, 2017.

[158] Francesco Romano, Gabriele Nava, Morteza Azad, Jernej Čamernik, Stefano Dafarra, Oriane Dermy, Claudia Latella, Maria Lazzaroni, Ryan Lober, Marta Lorenzini, et al. The codyco project achievements and beyond: Toward human aware whole-body controllers for physical human robot interaction. *IEEE Robotics and Automation Letters*, 3(1):516–523, 2018.

[159] Francisco Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. *Image and Vision Computing*, 76, 06 2018.

[160] Yeon Geol Ryu, Hyun Chul Roh, Si Jong Kim, Kwang Ho An, and Myung Jin Chung. Digital image stabilization for humanoid eyes inspired by human VOR system. In *IEEE Int. Conf. on Robotics and Biomimetics*, Dec 2009.

[161] J. Salini, V. Padois, and P. Bidaud. Synthesis of complex humanoid whole-body behavior: A focus on sequencing and tasks transitions. In *IEEE International Conference on Robotics and Automation*, pages 1283–1290, May 2011.

[162] Stefan Schaal. *Adaptive Motion of Animals and Machines*, chapter Dynamic Movement Primitives - A Framework for Motor Control in Humans and Humanoid Robotics, pages 261–280. Springer Tokyo, Tokyo, 2006.

[163] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A transport protocol for real-time applications. STD 64, RFC Editor, July 2003. `http://www.rfc-editor.org/rfc/rfc3550.txt`.

[164] Nicola Scianca, Marco Cognetti, Daniele De Simone, Leonardo Lanari, and Giuseppe Oriolo. Intrinsically stable MPC for humanoid gait generation. In *IEEE-RAS International Conference on Humanoid Robots*, pages 101–108, 2016.

[165] Nicola Scianca, Daniele De Simone, Leonardo Lanari, and Giuseppe Oriolo. Mpc for humanoid gait generation: Stability and feasibility. *IEEE Transactions on Robotics*, page 1–18, 2020.

[166] Mario Selvaggio, Jonathan Cacace, Claudio Pacchierotti, Fabio Ruggiero, and Paolo Robuffo Giordano. A shared-control teleoperation architecture for nonprehensile object transportation. *IEEE Transactions on Robotics*, 38(1):569–583, 2022.

[167] Mario Selvaggio, Jonathan Cacace, Claudio Pacchierotti, Fabio Ruggiero, and Paolo Robuffo Giordano. A shared-control teleoperation architecture for nonprehensile object transportation. *IEEE Transactions on Robotics*, 38(1):569–583, 2022.

[168] Bruno Siciliano, Lorenzo Sciavicco, Villani Luigi, and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2011.

[169] Joao Silvério, Sylvain Calinon, Leonel Dario Rozo, and Darwin G. Caldwell. Learning task priorities from demonstrations. *IEEE Transactions on Robotics*, 35:78–94, 2019.

[170] Gaganpreet Singh, Sergi Bermúdez i Badia, Rodrigo Ventura, and José Luís Silva. Physiologically attentive user interface for robot teleoperation: real time emotional state estimation and interface modification using physiology, facial expressions and eye movements. In *Int. Joint Conf. on Biomedical Engineering Systems and Technologies*, pages 294–302, Jan 2018.

[171] H. Singh, M. Panzirsch, A. Coelho, and C. Ott. Proxy-based approach for position synchronization of delayed robot coupling without sacrificing performance. *IEEE Robotics and Automation Letters*, 5(4):6599–6606, 2020.

[172] Ines Sorrentino, Francisco Javier Andrade Chavez, Claudia Latella, Luca Fiorio, Silvio Traversaro, Lorenzo Rapetti, Yeshasvi Tirupachuri, Nuno Guedelha, Marco Maggiali, Simeone Dussoni, et al. A novel sensorised insole for sensing feet pressure distributions. *Sensors*, 20(3):747, 2020.

[173] Didier Staudenmann, Karin Roeleveld, Dick F Stegeman, and Jaap H Van Dieën. Methodological aspects of semg recordings for force estimation–a tutorial and review. *J. of electromyography and kinesiology*, 20(3):375–387, 2010.

[174] Yusuke Sugano and Andreas Bulling. Self-calibrating head-mounted eye trackers using egocentric visual saliency. In *ACM Symp. on User Interface Software & Technology*, pages 363–372, Nov 2015.

[175] S. Tachi, K. Mlnamlzawa, M. Furukawa, and C. L. Fernando. Haptic media construction and utilization of human-harmonized "tangible" information environment. In *Int. Conf. on Artificial Reality and Telexistence*, pages 145–150, Dec 2013.

[176] Susumu Tachi. *Telexistence*. World Scientific 2nd edition, 2015.

[177] Susumu Tachi. Forty Years of Telexistence —From Concept to TELESAR VI (Invited Talk). In Yasuaki Kakehi and Atsushi Hiyama, editors, *ICAT-EGVE 2019 - International Conference on Artificial Reality and Telexistence and Eurographics Symposium on Virtual Environments*. Eurographics Association, 2019-09-11.

[178] Jukka Takala, Päivi Hämäläinen, Kaija Leena Saarela, Loke Yoke Yun, Kathiresan Manickam, Tan Wee Jin, Peggy Heng, Caleb Tjong, Lim Guan Kheng, Samuel Lim, and Gan Siok Lin. Global estimates of the burden of injury and illness at work in 2012. *Journal of Occupational and Environmental Hygiene*, 11(5):326–337, 2014.

[179] Toru Takenaka, Takashi Matsumoto, and Takahide Yoshiike. Real time motion generation and control for biped robot-1st report: Walking gait pattern generation. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1084–1091, Oct 2009.

[180] Mingxing Tan, Ruoming Pang, and Quoc V. Le. EfficientDet: Scalable and efficient object detection. In *in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 10778–10787, June 2020.

[181] **L. Penco**, B. Clément, V. Modugno, E.M. Hoffman, G. Nava, D. Pucci, N.G. Tsagarakis, J.-B. Mouret, and S. Ivaldi. Robust real-time whole-body motion retargeting from human to humanoid. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 425–432. IEEE, 2018.

[182] **L. Penco**, E. M. Hoffman, V. Modugno, W. Gomes, J. Mouret, and S. Ivaldi. Learning robust task priorities and gains for control of redundant robots. *IEEE Robotics and Automation Letters*, 5(2):2626–2633, 2020.

[183] **L. Penco**, J.-B. Mouret, and S. Ivaldi. Prescient teleoperation of humanoid robots. *CoRR*, abs/2107.01281, 2021.

[184] **L. Penco**, N. Scianca, V. Modugno, L. Lanari, G. Oriolo, and S. Ivaldi. A multimode teleoperation framework for humanoid loco-manipulation: An application for the iCub robot. *IEEE Robotics and Automation Magazine*, 26(4):73–82, 2019.

[185] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 23–30, Sept 2017.

[186] David Valenzuela-Urrutia, Rodrigo Muñoz-Riffo, and Javier Ruiz del Solar. Virtual reality-based time-delayed haptic teleoperation using point cloud data. *Journal of Intelligent & Robotic Systems*, 96:387–400, 2019.

[187] Darryl Veitch, Satish Babu, and Attila Pàsztor. Robust synchronization of software clocks across the internet. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, IMC '04, page 219–232, New York, NY, USA, 2004. Association for Computing Machinery.

[188] L. Vianello, **L. Penco**, W. Gomes, Y. You, S.M. Anzalone, P. Maurice, V. Thomas, and S. Ivaldi. Human-humanoid interaction and cooperation: a review. *Current Robotics Reports*, 2:441–454, 2021.

[189] Ruben Villegas, Jimei Yang, Duygu Ceylan, and Honglak Lee. Neural kinematic networks for unsupervised motion retargeting. In *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pages 8639–8648, Jun 2018.

[190] Christopher R Wagner, Susan J Lederman, and Robert D Howe. Design and performance of a tactile shape display using RC servomotors. *Haptics-e*, 3(4):1–6, 2004.

[191] Michael L. Walters, Dag S. Syrdal, Kerstin Dautenhahn, René te Boekhorst, and Kheng Lee Koay. Avoiding the uncanny valley: robot appearance, personality and consistency of behavior in an attention-seeking home scenario for a robot companion. *Autonomous Robots*, 24(2):159–178, Feb 2008.

[192] A. Wang, J. Ramos, J. Mayo, W. Ubellacker, J. Cheung, and S. Kim. The HERMES humanoid system: A platform for full-body teleoperation with balance feedback. In *IEEE/RAS Int. Conf. on Humanoid Robots*, pages 730–737, Nov 2015.

[193] Jilong Wang, Chunhong Lu, and Kun Zhang. Textile-based strain sensor for human motion detection. *Energy & Environmental Materials*, 3(1):80–100, 2020.

[194] Pierre-Brice Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *IEEE-RAS International Conference on Humanoid Robots*, pages 137–142, 2006.

[195] Pierre-brice Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 137–142, 2006.

[196] X. Xu and E. Steinbach. Elimination of cross-dimensional artifacts in the multi-dof time domain passivity approach for time-delayed teleoperation with haptic feedback. In *IEEE World Haptics Conference*, pages 223–228, 2019.

[197] A. Zamparelli, N. Scianca, L. Lanari, and G. Oriolo. Humanoid gait generation on uneven ground using Intrinsically Stable MPC. In *12th IFAC Symposium on Robot Control*, 2018.

[198] Xuan Zhao, Sakmongkon Chumkamon, Shuanda Duan, Juan Rojas, and Jia Pan. Collaborative human-robot motion generation using LSTM-RNN. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 1–9, 2018.

[199] Matt Zucker, Sungmoon Joo, Michael Grey, Christopher Rasmussen, Eric Huang, Michael Stilman, and Aaron Bobick. A general-purpose system for teleoperation of the DRC-HUBO humanoid robot. *J. of Field Robotics*, 32, 2015.

A

Prescient Teleoperation of Humanoid Robots

– Figure B.1. Reaching a bottle with compensation of a round-trip delay around 1.5s.

– Figure B.2. Teleoperation with compensation of a round-trip delay around 200ms.

– Figure B.3. Teleoperation with compensation of a round-trip delay around 500ms.

– Figure B.4. Teleoperation with compensation of a round-trip delay around 1s.

– Figure B.5. Teleoperation with compensation of a round-trip delay around 2s.

– Figure B.6. Teleoperation under unexpected human behaviors.

– Figure B.7. Comparison between the compensated trajectory and the ideal (non-delayed) trajectory with a round-trip delay around 1.5 s.

– Figure B.8. Comparison between the compensated trajectory and the ideal (non-delayed) trajectory with a round-trip delay around 2 s.

– Figure B.9. (Conforming the teleoperation to the intended motion) Prediction update according to observations.

– Figure B.10. (Conforming the teleoperation to the intended motion) Comparison between the compensated trajectory and the ideal (non-delayed) trajectory with a round-trip delay around 1.5s.

– Figure B.11. (Conforming the teleoperation to the intended motion) Prescient teleoperation with no delay.

– Figure B.12. (Conforming the teleoperation to new goals) Prediction update according to observations.

– Figure B.13. (Conforming the teleoperation to new goals) Comparison between the compensated trajectory and the ideal (non-delayed) trajectory with a round-trip delay around 1.5s.

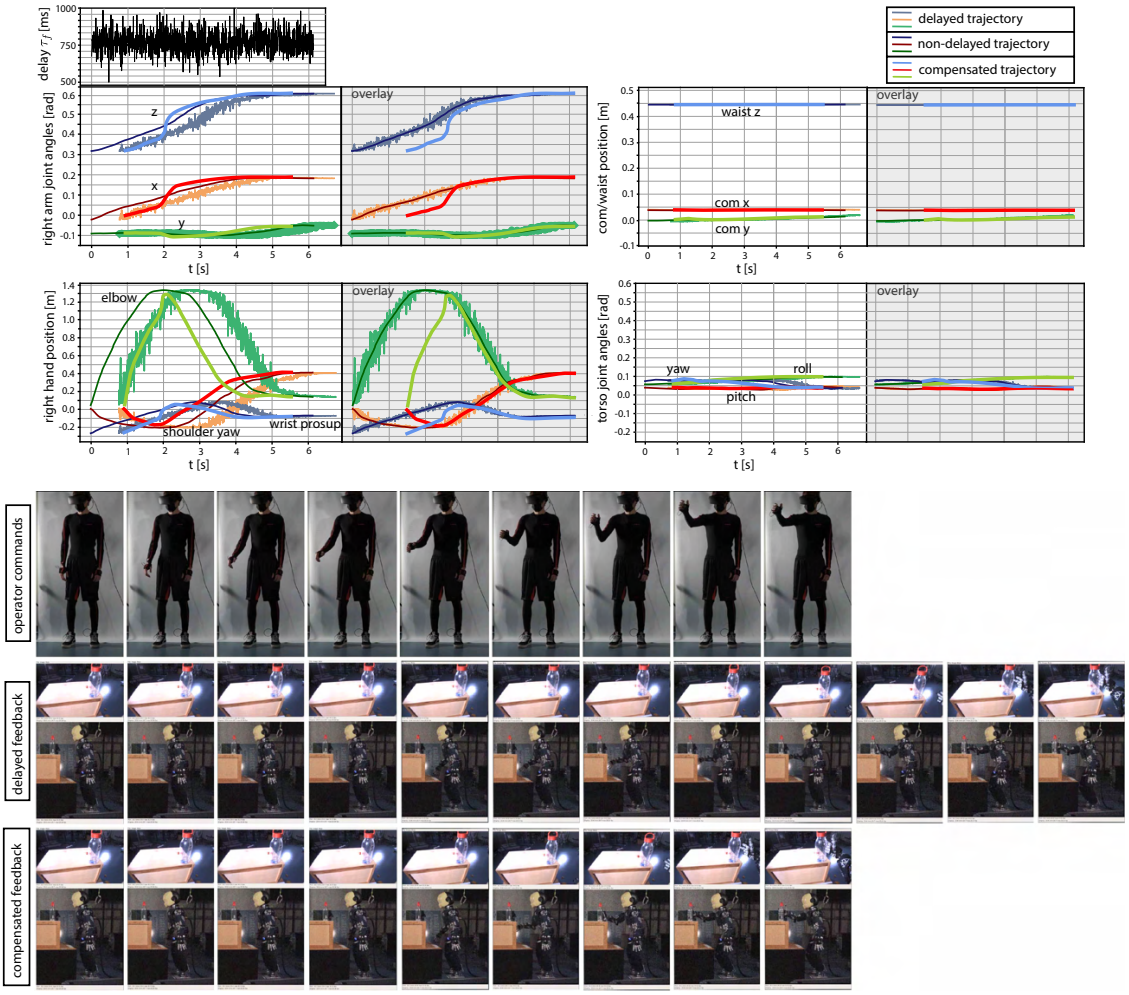


Figure A.1: Reaching a bottle with compensation of a round-trip delay around 1.5s. The robot is reaching a bottle located on top of a box. The forward delay is around 750ms with a jitter of 300ms (it follows a normal distribution with 750ms as mean and 100ms as standard deviation), while the backward delay is 750ms. The compensated trajectories (light red-green-blue lines) are the robot reference trajectories. These, first follow the delayed teleoperated signals (orange-teal-grey lines). Then, when the prediction is available, they anticipate the teleoperated motion (dark-colored lines) so to get a visual feedback at the user side coherent with what the operator is doing

.

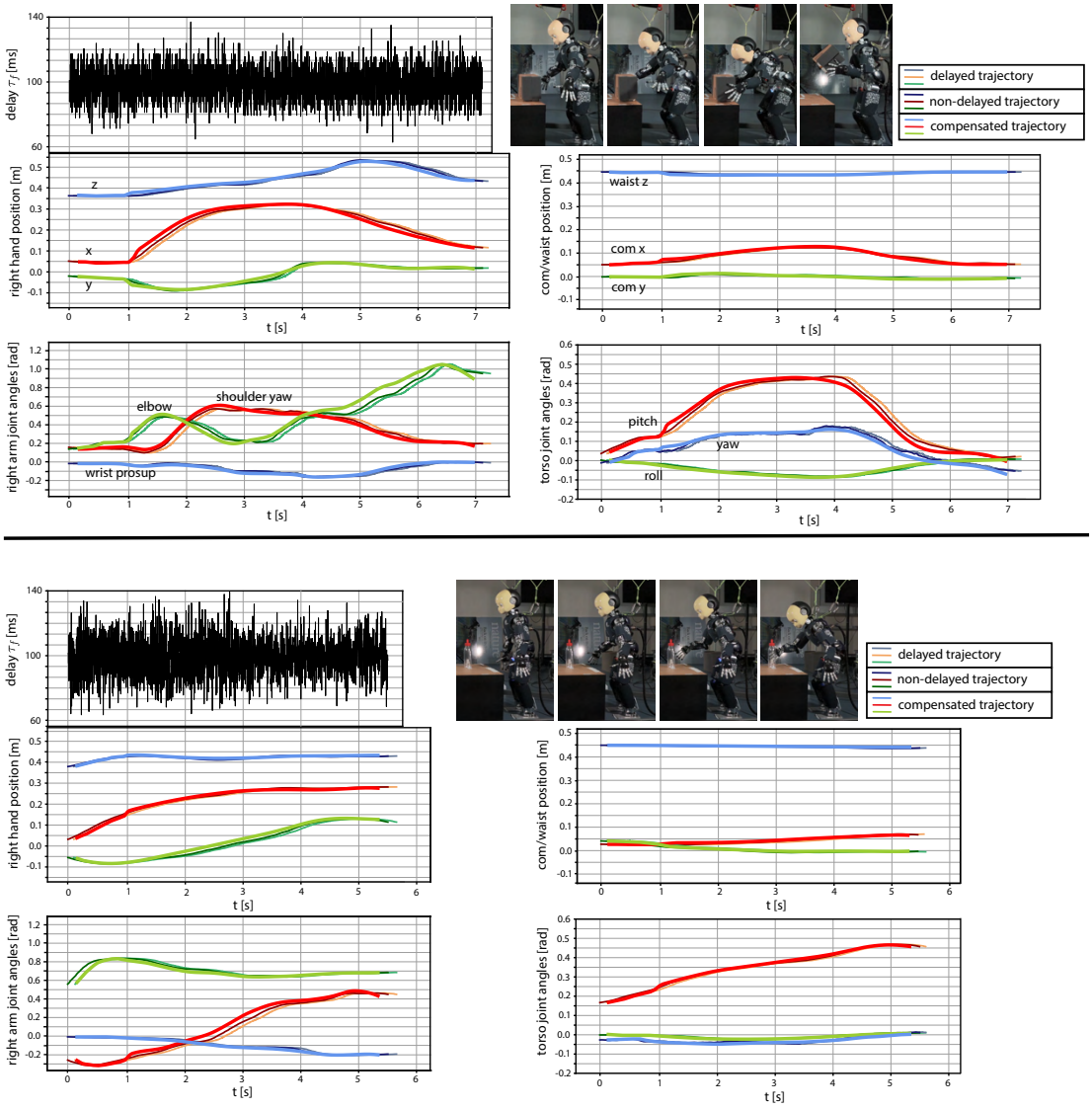


Figure A.2: Teleoperation with compensation of a round-trip delay around 200ms. Top: The robot is picking up a box located on a table. Bottom: The robot is reaching a bottle located on a table. The forward delay follows a normal distribution with 100ms as mean and 13.33ms as standard deviation, while the backward delay is 100ms. The compensated trajectories (light red-green-blue lines) are the robot reference trajectories. These, first follow the delayed teleoperated signals (orange-teal-grey lines). Then, when the prediction is available, they anticipate the teleoperated motion (dark-colored lines) so to get a visual feedback at the user side coherent with what the operator is doing

.

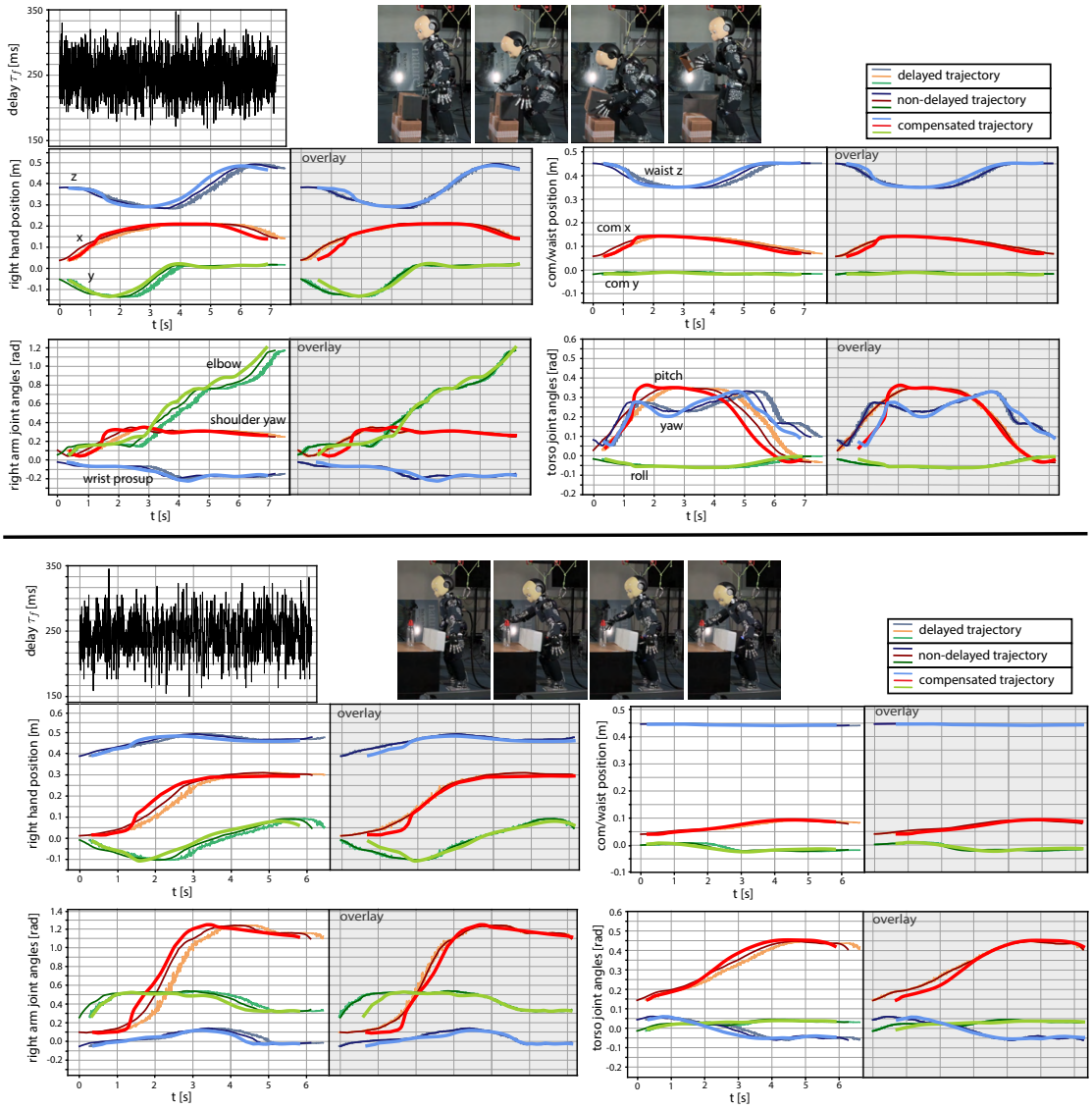


Figure A.3: Teleoperation with compensation of a round-trip delay around 500ms. Top: The robot is picking up a box in front of it at a low height. Bottom: The robot is reaching a bottle located on the table behind an obstacle. The forward delay follows a normal distribution with 250ms as mean and 33.33ms as standard deviation, while the backward delay is 250ms. The compensated trajectories (light red-green-blue lines) are the robot reference trajectories. These, first follow the delayed teleoperated signals (orange-teal-grey lines). Then, when the prediction is available, they anticipate the teleoperated motion (dark-colored lines) so to get a visual feedback at the user side coherent with what the operator is doing

.

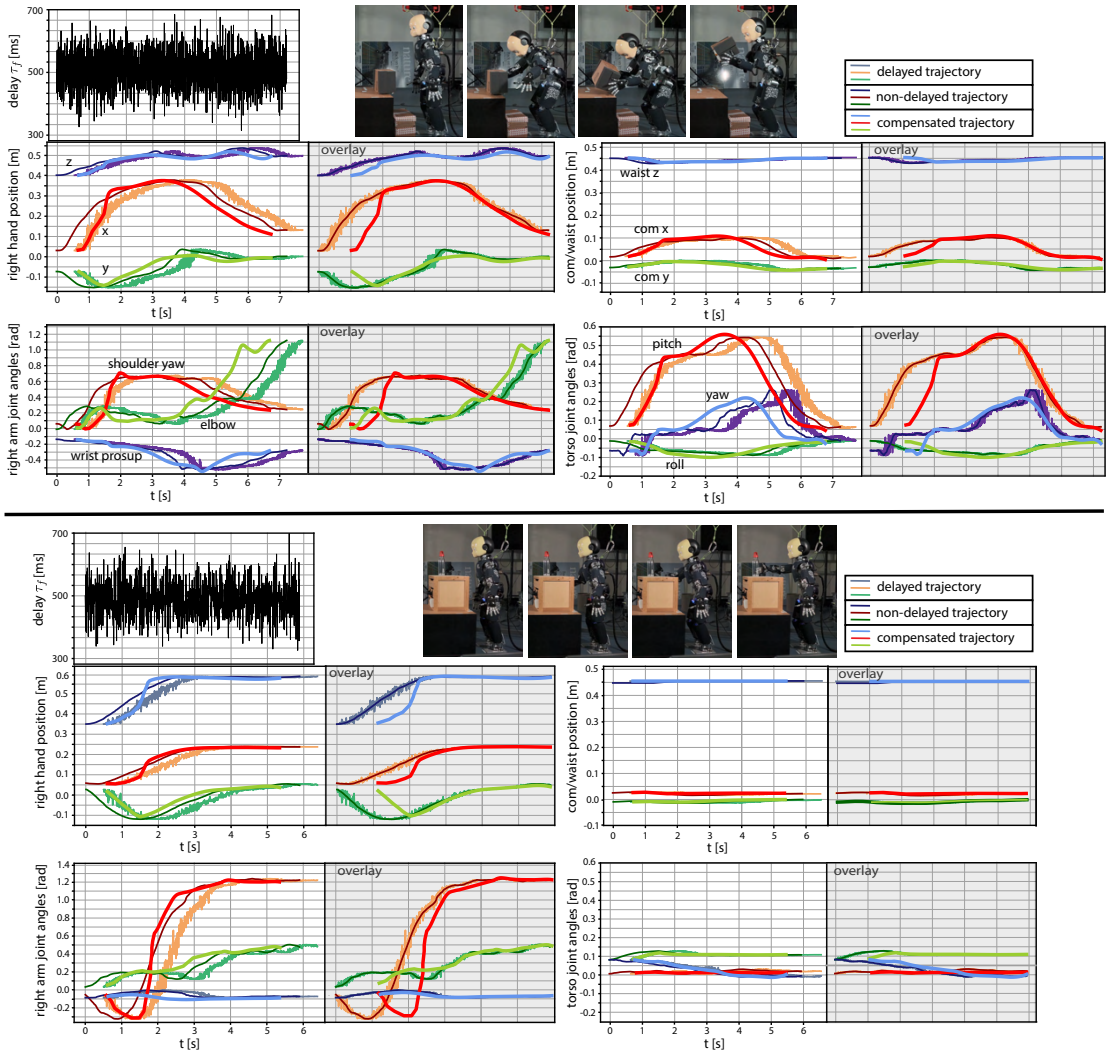


Figure A.4: Teleoperation with compensation of a round-trip delay around 1s. Top: The robot is picking up a box located on a table. Bottom: The robot is reaching a bottle located on a box, on the table. The forward delay follows a normal distribution with 500ms as mean and 66.67ms as standard deviation, while the backward delay is 500ms. The compensated trajectories (light red-green-blue lines) are the robot reference trajectories. These, first follow the delayed teleoperated signals (orange-teal-grey lines). Then, when the prediction is available, they anticipate the teleoperated motion (dark-colored lines) so to get a visual feedback at the user side coherent with what the operator is doing

.

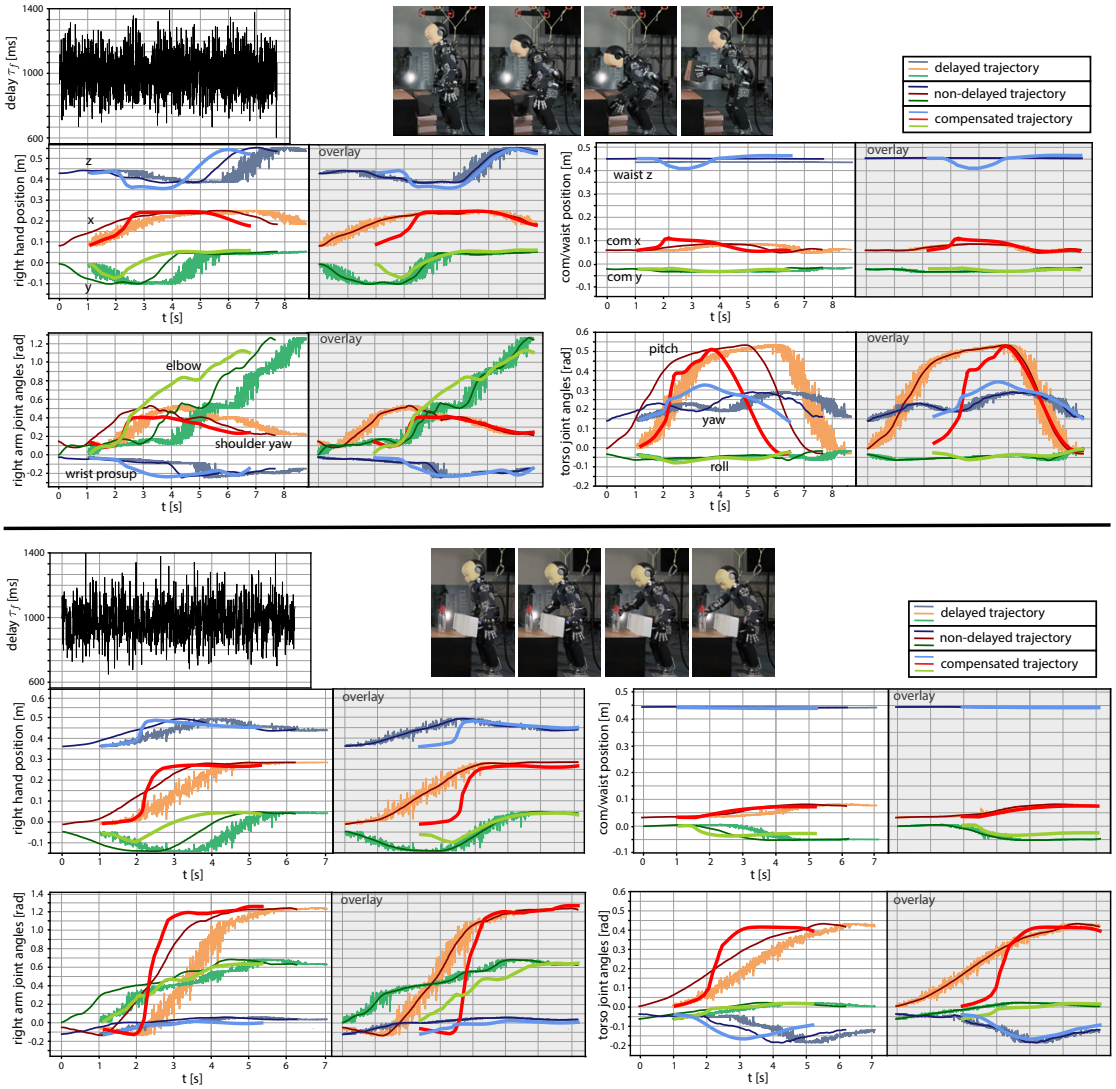


Figure A.5: Teleoperation with compensation of a round-trip delay around 2s. Top: The robot is picking up a box in front of it at a low height. Bottom: The robot is reaching a bottle located on the table behind an obstacle. The forward delay follows a normal distribution with 1000ms as mean and 133.33ms as standard deviation), while the backward delay is 1000ms. The compensated trajectories (light red-green-blue lines) are the robot reference trajectories. These, first follow the delayed teleoperated signals (orange-teal-grey lines). Then, when the prediction is available, they anticipate the teleoperated motion (dark-colored lines) so to get a visual feedback at the user side coherent with what the operator is doing

.

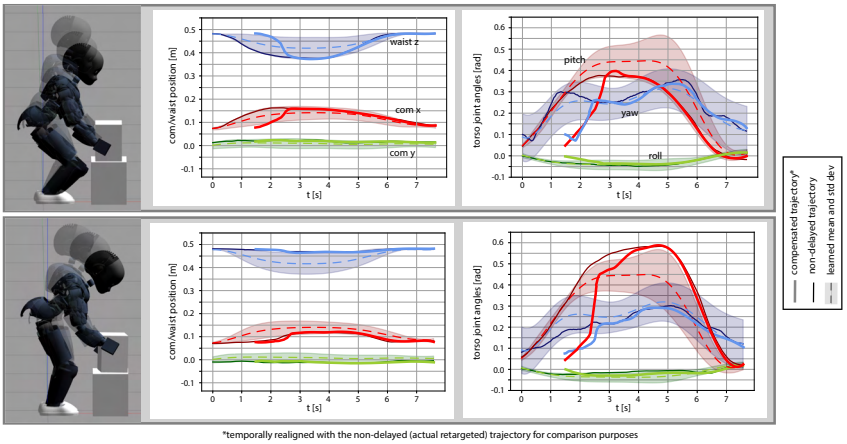


Figure A.6: Comparison between the compensated trajectory and the ideal (non-delayed) trajectory with a round-trip delay around 1.5 s. After the initial recognition period, our approach makes the robot follow the specific way the human is performing the task despite the delay. This is not the same as following the mean of previously demonstrated motions (here, the dashed line).

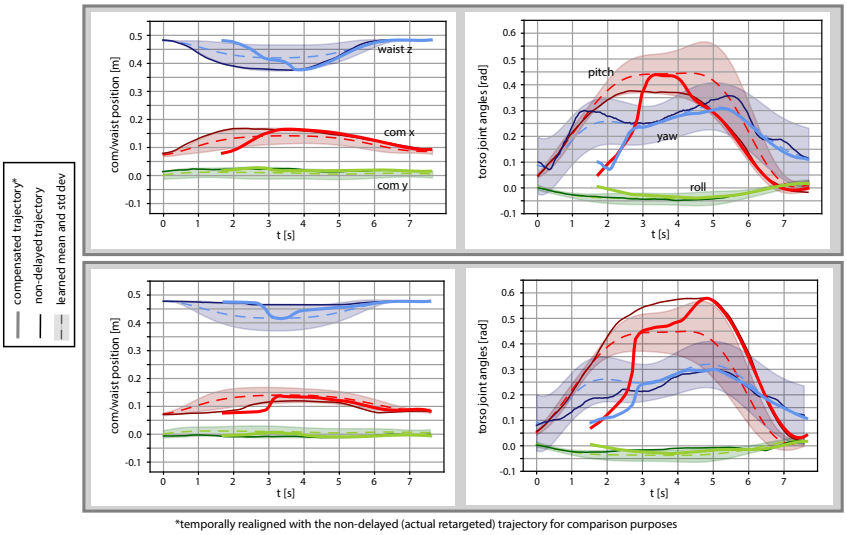


Figure A.7: Comparison between the compensated trajectory and the ideal (non-delayed) trajectory with a round-trip delay around 2 s. For very high delays even the continuous update of the prediction might not be sufficient to make the robot follow the specific way the operator is performing the task. Here, first the robot follows the delayed retargeted movements; then tends to follow the learned mean trajectory during the beginning of the compensation; only toward the end is able to compensate for the delay while following the actual retargeted reference.

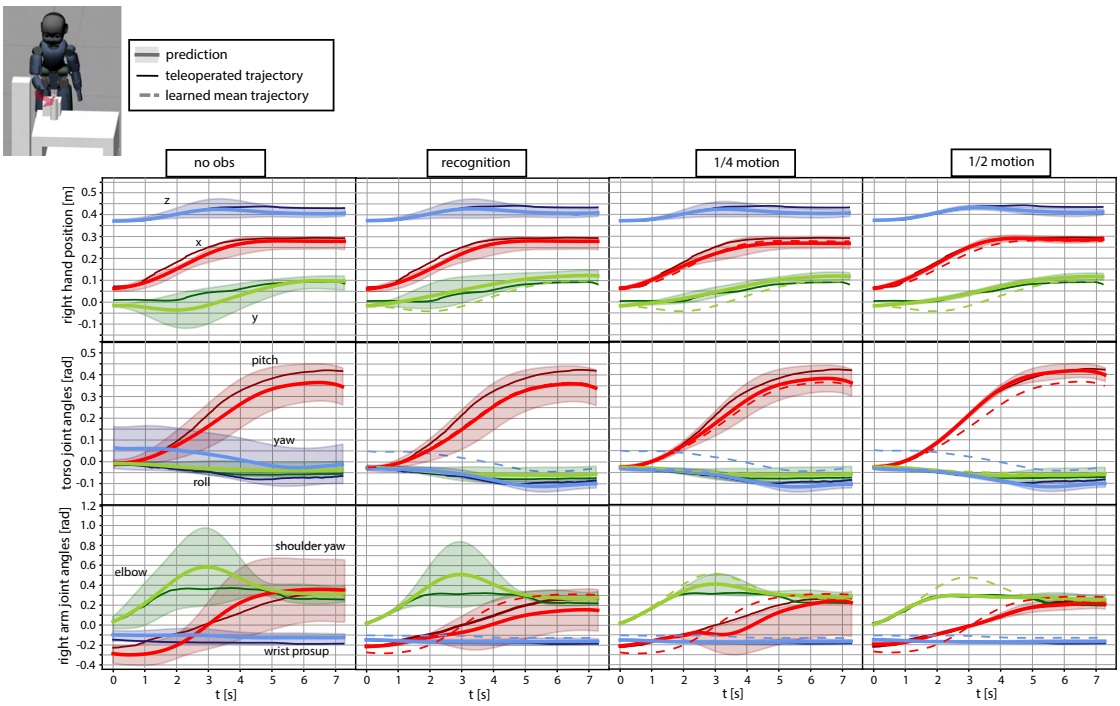


Figure A.8: (Conforming the teleoperation to the intended motion) Prediction update according to observations. The most relevant predicted trajectories (light colored lines) are compared to the non-delayed trajectories at the operator's side (dark colored lines), after observing different portions of the motion; a perfect prediction would mean that the light line (green/blue/red) line matches the dark line (green/blue/red). The non-delayed trajectories are from the testing set from Figure 5.9. From left to right, the figure shows the prediction given by the ProMPs learned from the demonstrations, the prediction updated after observing the first portion of motion used to infer the task and its duration, the prediction updated after observing a fourth of the motion, and after observing half of the motion.

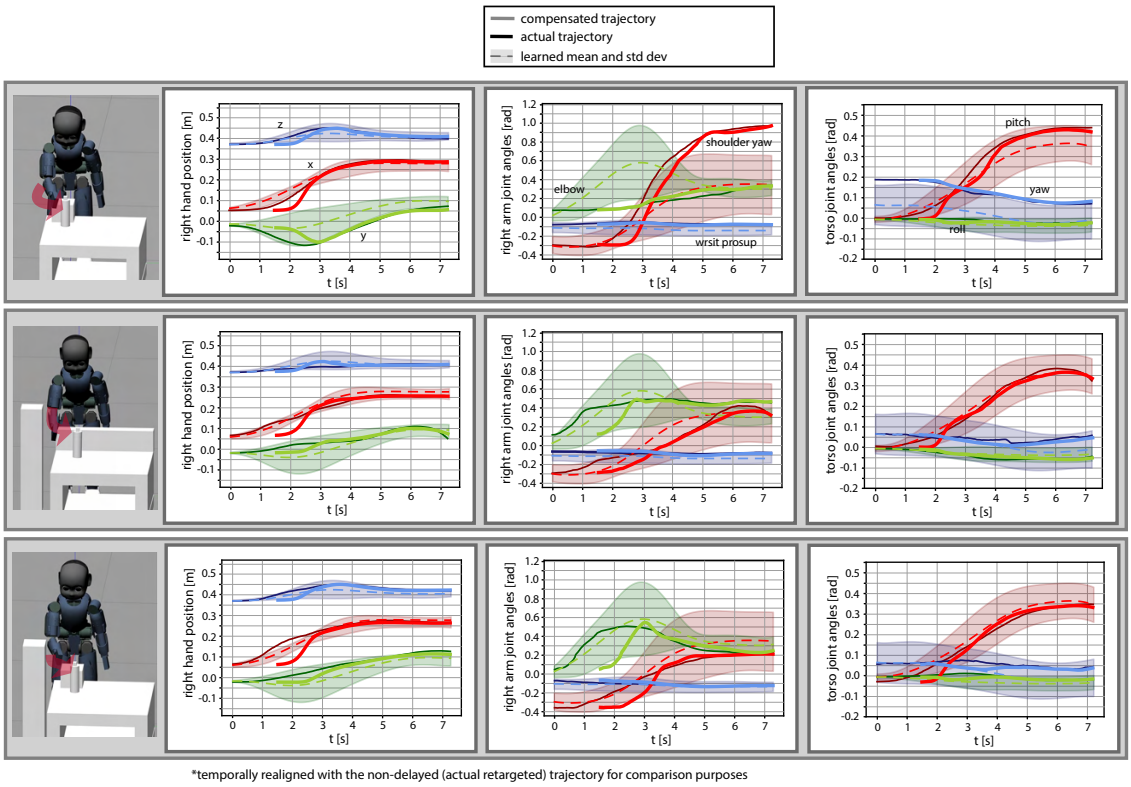


Figure A.9: (Conforming the teleoperation to the intended motion) Comparison between the compensated trajectory and the ideal (non-delayed) trajectory with a round-trip delay around 1.5s. On the top row, there is an unexpected obstacle (a small box) to avoid and the operator approaches the object by the right; on the second row, the obstacle in front of the robot is different; on the bottom row, there is the same obstacle from the top row with in addition a large obstacle on the right of the robot, which forces the operator to move the hand in between the two obstacles. These situations were not in the training set. After the initial recognition period, our approach makes the robot follow the specific way the human is performing the task despite the delay, and even if the robot is asked to perform the task in a way that has not been demonstrated before (but included in the distribution of the demonstrations). This is not the same as following the mean of previously demonstrated motions (here, the dashed line) or letting the robot replicate previously demonstrated motions. The non-delayed trajectories are some of the test trajectories from Figure 5.9, where the robot has to reach the bottle on the table in the presence of different obstacles that were not considered during the training.

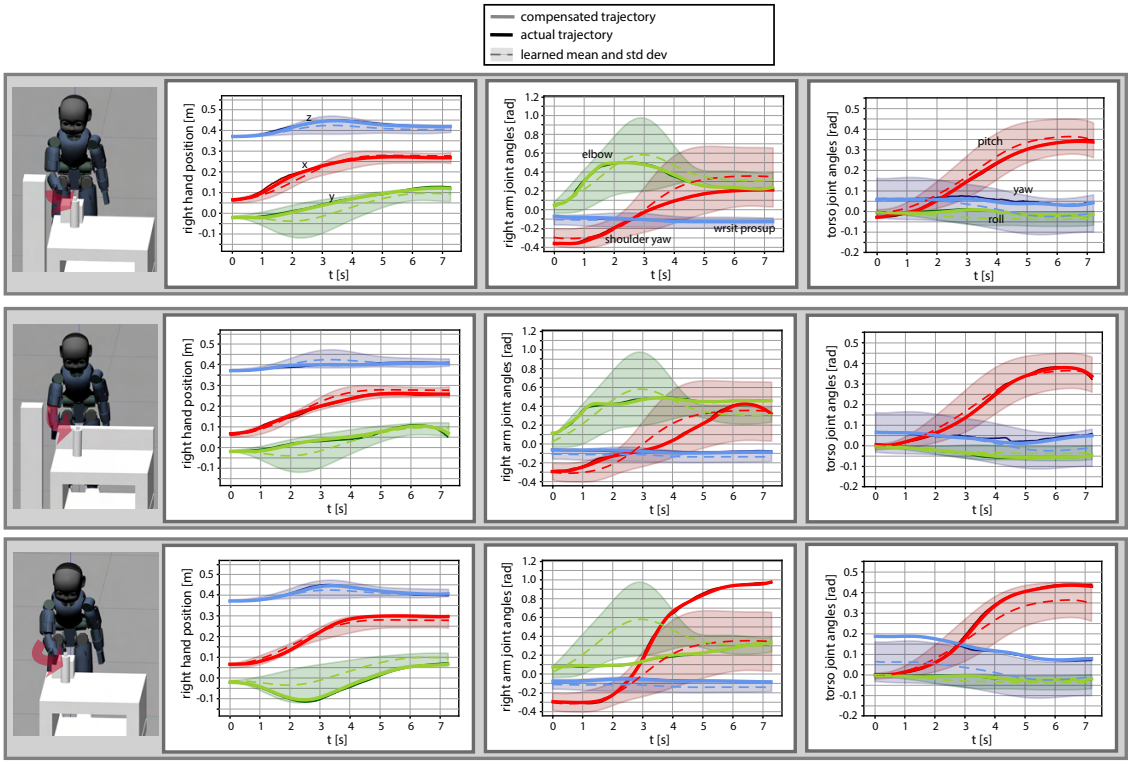


Figure A.10: (Conforming the teleoperation to the intended motion) Prescient teleoperation with no delay. The compensated trajectories resulting from our approach match almost perfectly the actual retargeted human references, when there is no delay. The actual trajectories are some of the test trajectories from Figure 5.9.

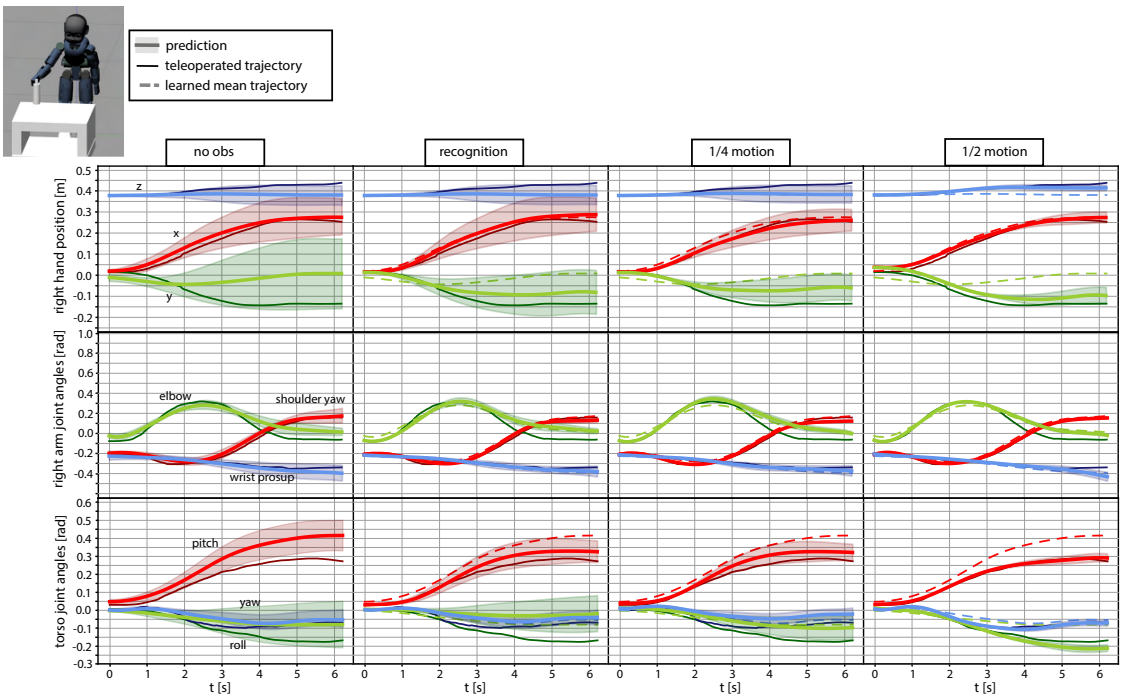


Figure A.11: (Conforming the teleoperation to new goals) Prediction update according to observations. The most relevant predicted trajectories (light colored lines) are compared to the non-delayed trajectories at the operator's side (dark colored lines), after observing different portions of the motion; a perfect prediction would mean that the light line (green/blue/red) line matches the dark line (green/blue/red). The non-delayed trajectories are from the testing set from Figure5.11. From left to right, the figure shows the prediction given by the ProMPs learned from the demonstrations, the prediction updated after observing the first portion of motion used to infer the task and its duration, the prediction updated after observing a fourth of the motion, and after observing half of the motion.

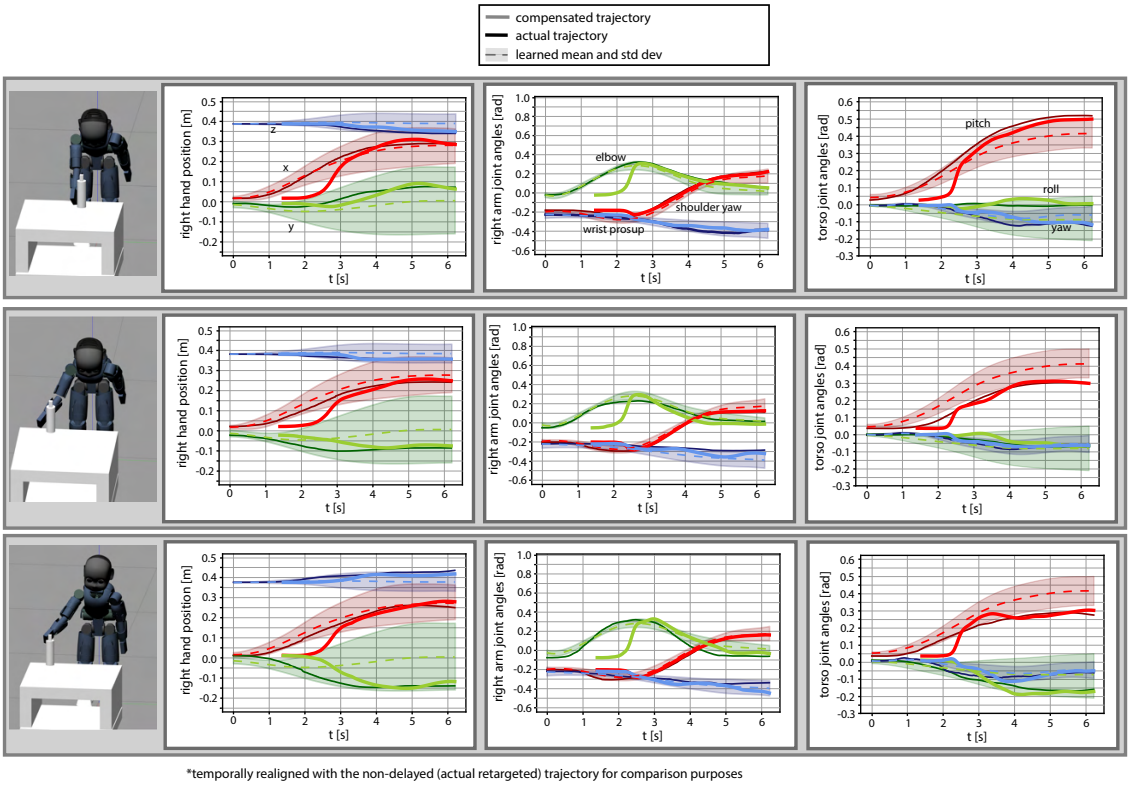


Figure A.12: (Conforming the teleoperation to new goals) Comparison between the compensated trajectory and the ideal (non-delayed) trajectory with a round-trip delay around 1.5s. The bottles are located on the table in different positions that were not in the training set (but included in the distribution of the demonstrations). The non-delayed trajectories are some of the test trajectories from Figure 5.11, where the robot has to reach the bottle on the table in the presence of different obstacles that were not considered during the training.

Résumé

Cette thèse vise à étudier des systèmes et des outils pour la télé-opération d'un robot humanoïde. La téléopération de robots est cruciale pour envoyer et contrôler les robots dans des environnements dangereux ou inaccessibles pour les humains (par exemple, des scénarios d'intervention en cas de catastrophe, des environnements contaminés ou des sites extraterrestres). Le terme télé-opération désigne le plus souvent le contrôle direct et continu d'un robot. Dans ce cas, l'opérateur humain guide le mouvement du robot avec son propre mouvement physique ou via un dispositif de contrôle. L'un des principaux défis est de contrôler le robot de manière à garantir son équilibre dynamique tout en essayant de suivre les références humaines. De plus, l'opérateur humain a besoin d'un retour d'information sur l'état du robot et de son site via des capteurs à distance afin d'appréhender la situation ou de se sentir physiquement présent sur le site, produisant des comportements de robot efficaces. Des complications surviennent lorsque le réseau de communication n'est pas idéal. Dans ce cas, les commandes de l'homme au robot ainsi que la rétroaction du robot à l'homme peuvent être retardées. Ces délais peuvent être très gênants pour l'opérateur humain, qui ne peut pas télé-opérer efficacement son avatar robotique.

Un autre point crucial à considérer lors de la mise en place d'un système de télé-opération est le grand nombre de paramètres qui doivent être réglés pour contrôler efficacement les robots télé-opérés. Des approches d'apprentissage automatique et des optimiseurs stochastiques peuvent être utilisés pour automatiser l'apprentissage de certains paramètres.

Dans cette thèse, nous avons proposé un système de télé-opération qui a été testé sur le robot humanoïde iCub. Nous avons utilisé une combinaison de capture de mouvement basée sur la technologie inertielle comme périphérique de contrôle pour l'humanoïde et un casque de réalité virtuelle connecté aux caméras du robot pour obtenir un retour visuel. Nous avons d'abord traduit les mouvements humains en mouvements robotiques équivalents en développant une approche de retargeting de mouvement qui atteint la ressemblance humaine tout en essayant d'assurer la faisabilité du mouvement transféré. Nous avons ensuite implémenté un contrôleur du corps entier pour permettre au robot de suivre le mouvement humain reciblé. Le contrôleur a ensuite été optimisé en simulation pour obtenir un bon suivi des mouvements de référence du corps entier, en recourant à un optimiseur stochastique multi-objectifs, ce qui nous a permis de trouver des solutions robustes fonctionnant sur le robot réel en quelques essais.

Pour télé-opérer les mouvements de marche, nous avons implémenté un mode de télé-opération de niveau supérieur dans lequel l'utilisateur peut utiliser un joystick pour envoyer des commandes de référence au robot. Nous avons intégré ce paramètre dans le système de télé-opération, ce qui permet à l'utilisateur de basculer entre les deux modes différents.

Un problème majeur empêchant le déploiement de tels systèmes dans des applications réelles est la présence de retards de communication entre l'entrée humaine et le retour du robot: même quelques centaines de millisecondes de retard peuvent irrémédiablement perturber l'opérateur, encore plus quelques secondes. Pour surmonter ces retards, nous avons introduit un système dans lequel un robot humanoïde exécute des commandes avant de les recevoir, de sorte que le retour visuel semble être synchronisé avec l'opérateur, alors que le robot exécutait les commandes dans le passé. Pour ce faire, le robot prédit en permanence les commandes futures en interrogeant un modèle d'apprentissage automatique formé sur les trajectoires passées et conditionné aux dernières commandes reçues.

Mots-clés: Télé-opération, robotique humanoïde, retargeting de mouvement, optimisation de contrôl, compensation des retards

Abstract

This thesis aims to investigate systems and tools for teleoperating a humanoid robot. Robot teleoperation is crucial to send and control robots in environments that are dangerous or inaccessible for humans (e.g., disaster response scenarios, contaminated environments, or extraterrestrial sites). The term teleoperation most commonly refers to direct and continuous control of a robot. In this case, the human operator guides the motion of the robot with her/his own physical motion or through some physical input device. One of the main challenges is to control the robot in a way that guarantees its dynamical balance while trying to follow the human references. In addition, the human operator needs some feedback about the state of the robot and its work site through remote sensors in order to comprehend the situation or feel physically present at the site, producing effective robot behaviors. Complications arise when the communication network is non-ideal. In this case the commands from human to robot together with the feedback from robot to human can be delayed. These delays can be very disturbing for the human operator, who cannot teleoperate their robot avatar in an effective way.

Another crucial point to consider when setting up a teleoperation system is the large number of parameters that have to be tuned to effectively control the teleoperated robots. Machine learning approaches and stochastic optimizers can be used to automate the learning of some of the parameters.

In this thesis, we proposed a teleoperation system that has been tested on the humanoid robot iCub. We used an inertial-technology-based motion capture suit as input device to control the humanoid and a virtual reality headset connected to the robot cameras to get some visual feedback. We first translated the human movements into equivalent robot ones by developping a motion retargeting approach that achieves human-likeness while trying to ensure the feasibility of the transferred motion. We then implemented a whole-body controller to enable the robot to track the retargeted human motion. The controller has been later optimized in simulation to achieve a good tracking of the whole-body reference movements, by recurring to a multi-objective stochastic optimizer, which allowed us to find robust solutions working on the real robot in few trials.

To teleoperate walking motions, we implemented a higher-level teleoperation mode in which the user can use a joystick to send reference commands to the robot. We integrated this setting in the teleoperation system, which allows the user to switch between the two different modes.

A major problem preventing the deployment of such systems in real applications is the presence of communication delays between the human input and the feedback from the robot: even a few hundred milliseconds of delay can irremediably disturb the operator, let alone a few seconds. To overcome these delays, we introduced a system in which a humanoid robot executes commands before it actually receives them, so that the visual feedback appears to be synchronized to the operator, whereas the robot executed the commands in the past. To do so, the robot continuously predicts future commands by querying a machine learning model that is trained on past trajectories and conditioned on the last received commands.

Keywords: Teleoperation, humanoid robotics, motion retargeting, control optimization, delay compensation.