HAL
open science

A dense and direct approach to visual servoing using depth maps

Céline Teulière, Eric Marchand

To cite this version:

Céline Teulière, Eric Marchand. A dense and direct approach to visual servoing using depth maps. IEEE Transactions on Robotics, 2014, 30 (5), pp.1242-1249. 10.1109/TRO.2014.2325991 . hal-00991641v2

HAL Id: hal-00991641

https://inria.hal.science/hal-00991641v2

Submitted on 28 Apr 2015

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.

A dense and direct approach to visual servoing using depth maps

Céline Teulière and Eric Marchand

***Abstract*—This paper presents a novel 3D servoing approach using dense depth maps to perform robotic tasks. With respect to position-based approaches, our method does not require the estimation of the 3D pose (direct), nor the extraction and matching of 3D features (dense) and only requires dense depth maps provided by 3D sensors. Our approach has been validated in various servoing experiments using the depth information from a low cost RGB-D sensor. Positioning tasks are properly achieved despite noisy measurements, even when partial occlusions or scene modifications occur. We also show that, in cases where a reference depth map cannot be easily available, synthetic ones generated with a rendering engine still lead to satisfactory positioning performances. Application of the approach to the navigation of a mobile robot is also demonstrated.**

***Index Terms*—dense sensor-based control, depth map, visual servoing.**

I. Introduction

Most of the robotic positioning tasks are still achieved today by estimating first the relative pose between the robot and the scene or the object of interest, and then using a pose-based control scheme, as initially proposed in [29]. However, the pose estimation problem itself is complex in its general formulation. Also known as the *3D localization problem* [17], this problem has been widely investigated by the computer vision community but remains non-trivial for vision sensors alone, in particular in low-textured environments. Using range data, a range flow formulation has been proposed [14][11] to estimate the 3D pose of a mobile robot. Alternatively, the alignment of successive 3D point clouds using ICP [2], [5] has become a very popular method. Many variants have been proposed in the literature [23] and the development of the so-called RGB-D cameras attracted a lot of attention on these methods in the recent years [27], [21], [13], [22]. Some work also consider crude global alignment using a global voting scheme within a transformed space [18].

In this paper, we propose to perform robotic tasks without reconstructing the full 3D pose between the robot and its environment, but using a sensor-based servoing scheme, the considered data being directly the depth map obtained from a range sensor. Our approach is thus related to other sensor-based methods, such as image-based visual servoing (IBVS) [3], where a robotic task is expressed directly as the regulation of a visual error. In IBVS, the visual error is usually defined as the difference between a current and a desired set of geometric features (points, straigth lines, etc.) selected from the image, to control the desired degrees of freedom. Therefore, IBVS schemes usually require the extraction of visual features from image measurements, and their matching in successive frames. However, those steps, based on image processing techniques, are often considered as the bottleneck of visual servoing methods. In the tracking literature, dense approaches that do not require matching have already been proposed, based on the "brightness constancy constraint" [12], [1], [25]. In [9] this constraint is used in a stereo system to track planes and is applied to mobile robot localization. Recently, some visual servoing work also proposed to use all the image directly, without any extraction or matching step, by minimizing the difference between the current image and a reference image. This approach is referenced as *photometric visual servoing* [7][6]. However, luminance-based approaches are not always applicable since they require texture, and stable lighting condition, or an accurate knowledge of the materials and light sources to model image formation [26].

In our work we propose to use the dense depth map obtained from a range sensor as a visual feature for positioning task wrt. non planar scenes, without any feature extraction or matching step. This is a major difference with respect to approaches such as [20] where 3D points have to be matched. We derive a new control law for a robot positioning or navigation using this feature directly. Our approach is thus both direct (without any 3D pose estimation) and dense (without feature extraction or matching). A first version of this work has been presented in [28]. We provide here an extended version, with new experimental results underlying the strong potential of this approach. To the best of our knowledge, this is the first work proposing such a dense depth-based visual servoing.

II. Direct dense depth map servoing

This section presents the heart of our approach, i.e. how to control a robot using dense depth maps. We first introduce what we call a depth map and what it means to use it as feature (Section II-A). Then we derive the fundamental equations necessary to compute our control law (Sections II-B and II-C).

A. Depth map sensing

There are multiple technologies of sensors capable of providing depth (or range) information. Most range sensors without contact are active, and based on the time of flight (ToF) principle: the idea is to send waves of known velocity and measure the time it takes them to go from the sensor and come back after reflection on the scene. This can be achieved by sending light pulses. Another approach consists in using a modulated signal and measuring the phase shift. In each case, the depth information is derived knowing the velocity of the sent signal (eg: Laser scans, sonars, radars, ToF or RGB-D cameras). Another existing technology for active range sensing is based on structured light: known patterns (stripes, dots, ...) are projected onto the scene and the depth information is deduced from their deformation. This technology is used for instance in the recent Microsoft Kinect or Asus Xtion pro devices, based on PrimeSense technology [10]. Depth can also be measured with passive sensors such as cameras: by matching image features in two different views of a calibrated stereo rig, depth can be computed from geometry. The depth information is sparse when a finite set of features are matched, but dense depth maps can also be obtained [24].

In the following, we consider a range sensor capable of providing dense depth maps. Without loss of generality, the range measurements are expressed in sensor centered cartesian coordinates. We also consider that the depth map is represented according to a perspective projection model (see Figure 1). This is a natural choice for any range sensor based on perspective cameras (stereo pairs, or structured light such as the Kinect sensor used in our experiments). It is also very general since any dense depth map coming from other sensors (laser, radar, etc.) can be converted with such a perspective projection. Formally, we denote by $Z(x, y, t)$ the depth at time t of the 3D point of coordinates (X, Y, Z) in the sensor frame, with $X = xZ$ and $Y = yZ$, (x, y) being the metric image coordinates. Figure 2 gives an example of depth map obtained from Microsoft Kinect RGB-D sensor, where the depth values have been scaled to greyscale levels. White pixels correspond to unavailable depth values, i.e. pixels where the sensor could not compute any depth information. Note also that for better visualisation purpose, we applied histogram equalization

C. Teulière was with Inria, Lagadic Project, Rennes, France. She is now with the Blaise Pascal University, Pascal Institute, Clermont-Ferrand, France. e-mail: celine.teuliere@univ-bpclermont.fr

E. Marchand is with Université de Rennes 1, IRISA, Inria, Lagadic Project, Rennes, France. email: marchand@irisa.fr

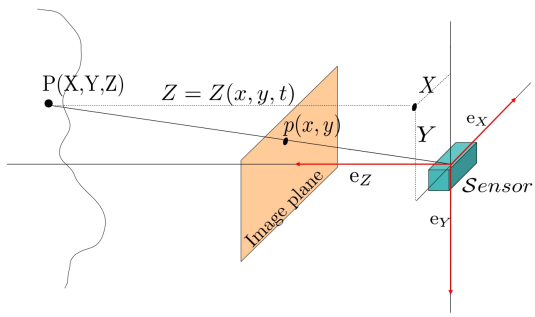
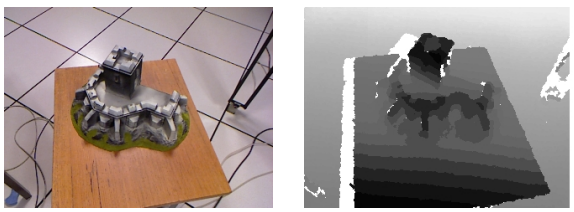


Fig. 1. Sensor frame representation.

on the depth maps shown throughout the paper, but the experiments use the depth map directly.



(a) (b)

Fig. 2. Example of static scene (a) and corresponding depth map representation (b) acquired from Microsoft Kinect sensor. The darkest pixels correspond to the smallest depths. White pixels correspond to unavailable data.

The next section shows how such dense depth maps can be used to control a robot.

B. Modeling

Let us consider that a robot end effector is equipped with a range sensor (Figure 3).

Fig. 3. ADEPT Viper robotic system equipped with a Microsoft Kinect sensor.

We express a positioning task as the regulation of the feature $\mathbf{Z}$ to a desired value $\mathbf{Z}^*$. Here, $\mathbf{Z} = (Z_1, ..., Z_N)$ is a vector containing the N depth values corresponding to the current dense depth map. The desired value $\mathbf{Z}^*$ thus corresponds to a reference depth map acquired at the desired robot pose.

Therefore, the control law to design aims at regulating the following error to zero:

$$\mathbf{e} = \mathbf{Z} - \mathbf{Z}^* = \begin{pmatrix} \vdots \\ Z_i - Z_i^* \\ \vdots \end{pmatrix} \tag{1}$$

An illustration of such an error is given in Figure 4.

In analogy with the visual servoing framework [3] we denote by $\mathbf{L_Z}$ the interaction matrix associated to the feature $\mathbf{Z}$, and characterized by the relation:

$$\frac{\partial \mathbf{Z}}{\partial t} = \mathbf{L_Z}\mathbf{v} \tag{2}$$

where $\frac{\partial \mathbf{Z}}{\partial t}$ is the temporal variation of the depth and $\mathbf{v} = (\boldsymbol{v}, \boldsymbol{\omega})$ is the sensor instantaneous velocity, with $\boldsymbol{v}$ the translational velocity, and $\boldsymbol{\omega}$ the rotational velocity.

We now derive the expression of the matrix $\mathbf{L_Z}$ which will be required in the control law (Section II-C). In the following, we consider the continuous formulation of the depth map as a surface $Z(x, y, t)$. Assuming that the scene is rigid and the surface $Z(x, y, t)$ is smooth[1], taking its full derivative leads to:

$$\dot{Z} = \frac{dZ}{dt} = \frac{\partial Z}{\partial x}\dot{x} + \frac{\partial Z}{\partial y}\dot{y} + \frac{\partial Z}{\partial t}, \tag{3}$$

where $(\dot{x}, \dot{y})$ is the 2D velocity of the image point (x, y). Equation (3) is known as the *range flow constraint equation* [30] or *elevation rate constraint equation* [14]. It is very similar to the *brightness change constraint equation* that is used in the computation of optical flow [15] and used in direct photometric visual servoing methods [6]. The main difference is that in the luminance case, an additional assumption is made to constrain the brightness time derivative to be zero.

From equation (3), the temporal variation of the depth is immediately deduced:

$$\frac{\partial Z}{\partial t} = \dot{Z} - A\dot{x} - B\dot{y}, \tag{4}$$

where $A = \frac{\partial Z}{\partial x}$ and $B = \frac{\partial Z}{\partial y}$. Therefore, the interaction matrix $\mathbf{L}_Z$ related to one depth value is expressed by:

$$\mathbf{L}_Z = \mathbf{L}_{P_Z} - A\mathbf{L}_x - B\mathbf{L}_y. \tag{5}$$

The matrices $\mathbf{L}_x$, $\mathbf{L}_y$ defined such that $\dot{x} = \mathbf{L}_x\mathbf{v}$ and $\dot{y} = \mathbf{L}_y\mathbf{v}$ are the well-known interaction matrices of image point coordinates, given by [3]:

$$\mathbf{L}_x = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \end{bmatrix} \tag{6}$$

$$\mathbf{L}_y = \begin{bmatrix} 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix}, \tag{7}$$

and $\mathbf{L}_{P_Z}$ is the interaction matrix related to the coordinate Z of a 3D point, such that $\dot{Z} = \mathbf{L}_{P_Z}\mathbf{v}$. It is given by [3][20]:

$$\mathbf{L}_{P_Z} = \begin{bmatrix} 0 & 0 & -1 & -yZ & xZ & 0 \end{bmatrix}. \tag{8}$$

Finally, replacing (6), (7) and (8) in (5), we get:

$$\mathbf{L}_Z = \begin{bmatrix} \frac{A}{Z} & \frac{B}{Z} & -\frac{Z+xA+yB}{Z} & Z_{w_x} & Z_{w_y} & Z_{w_z} \end{bmatrix}, \tag{9}$$

where $Z_{w_x} = -yZ - xyA - (1+y^2)B$, $Z_{w_y} = xZ + (1+x^2)A + xyB$ and $Z_{w_z} = xB - yA$. Note that this expression underlines one of the main differences between our approach and sparse 3D approaches [20], as discussed in Appendix A. The full interaction matrix $\mathbf{L_Z}$ of size $N \times 6$ corresponding to the entire depth map is thus the stack of the 1×6 matrices $\mathbf{L}_{Z_i}$:

$$\mathbf{L_Z} = \begin{bmatrix} \mathbf{L}_{Z_1} \\ \vdots \\ \mathbf{L}_{Z_N} \end{bmatrix}. \tag{10}$$

C. Control law

We consider the following control law:

$$\mathbf{v} = -\lambda \mathbf{L}_{\mathbf{Z}}^{+} (\mathbf{Z} - \mathbf{Z}^*) \tag{11}$$

where λ is a scalar gain parameter and $\mathbf{L}_{\mathbf{Z}}^{+}$ denotes the pseudo-inverse of $\mathbf{L_Z}$ defined by $\mathbf{L}_{\mathbf{Z}}^{+} = (\mathbf{L_Z}^\top \mathbf{L_Z})^{-1}\mathbf{L_Z}^\top$. Note that exactly the same demonstration as for IBVS as given in [4] Section 24.3.4 allows

[1]The points in the surface where this assumption do not hold will be discarded by the M-estimation process presented in section III.

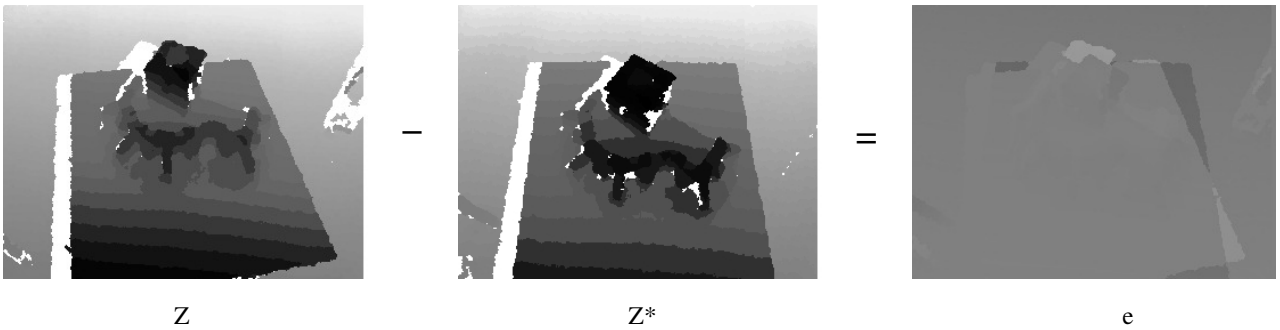


Fig. 4. The task error is the difference of depth maps $\mathbf{Z} - \mathbf{Z}^*$.

demonstrating the local asymptotic stability (LAS) of this control law under the condition that $\mathbf{L}_\mathbf{Z}^+\mathbf{L}_\mathbf{Z} > 0$ in the neighborhood of $\mathbf{Z} = \mathbf{Z}^*$. This condition is ensured as soon as $\mathbf{L}_\mathbf{Z}$ is of full rank 6 since $\mathbf{L}_\mathbf{Z}^+\mathbf{L}_\mathbf{Z} = \mathbf{I}_6$ in that case.

The interaction matrix given by (10) has to be of full rank for the system to be stable. For instance, a fully planar scene would lead to a rank 3 matrix where only 3 degrees of freedom (dof) could be controlled to form a plane-to-plane virtual linkage. The interpretation of this is that an infinite number of poses would lead to the same depth map perception. However the full-rank assumption is easily fulfilled in real world scenarios where depth variations are discriminative enough to avoid indetermination. Typically, observing points from 3 non parallel planes is sufficient.

III. Practical issues and robustness improvements

In the previous section, we presented our depth map based servoing method. When testing it, we found that this method was efficient in simulation sequences, with perfect data, but we had to face some practical issues in real conditions, in particular, in our case, using a Kinect sensor. This section presents the modifications we had to undertake in order to improve the robustness of the servoing task with respect to noisy and incomplete measurements (Section III-A) and to scene perturbations and occlusions (Section III-B).

A. Noisy and incomplete measurements

As illustrated in Figure 2-b the depth map acquired by a Kinect sensor is noisy and incomplete. In practice, we only considered the pixels for which a depth value was available both in the reference $\mathbf{Z}^*$ and the current $\mathbf{Z}$ depth maps. This means that the number N of depth values in $\mathbf{Z}$ and (10), is inferior to the size of the depth map. In the experiments presented in this paper, about 80% of the total number of pixels could typically be used.

In addition, we reduced the noise by applying a standard 3×3 Gaussian filter on the depth maps, the convolution being computed only with the valid neighbors.

Similarly, the spatial gradient was computed using a standard 3×3 derivative kernel taking into account the valid neighbors only.

B. Occlusions and scene modifications

Another issue to take into account is the possibility of partial occlusions or scene modifications during the servoing process. To reduce the effect of such events on the task achievement, we use robust M-estimation [16]. We thus introduce a modification of our task objective (1) allowing uncertain measures to be less likely considered or in some cases completely rejected. The new task error is given by [8]:

$$\mathbf{e} = \mathbf{D}(\mathbf{Z} - \mathbf{Z}^*) \tag{12}$$

where $\mathbf{D}$ is a diagonal weighting matrix: $\mathbf{D} = diag\,(w_1, ..., w_N)$. The new control law thus becomes:

$$\mathbf{v} = -\lambda(\mathbf{D}\mathbf{L}_\mathbf{Z})^+\mathbf{D}(\mathbf{Z} - \mathbf{Z}^*), \tag{13}$$

The weights w_i in $\mathbf{D}$ are computed using Tukey's estimator. The reader can refer to [16] for details on M-estimation.

IV. Experimental results

In this section we first provide the experimental validation of our approach for positioning tasks (Section IV-A). Then its application to the navigation of a mobile robot is proposed (Section IV-B).

A. Positioning tasks

In our positioning tasks, a Kinect sensor has been mounted on a ADEPT Viper robot (see Figure 3). In each experiment, the task is expressed as the minimization of the error (12) between a fixed desired depth map and the current one. The control law (13) is computed with a fixed gain $\lambda = 2.5$. The depth maps are acquired using the LibFreenect[2] driver through the ViSP library [19], with a resolution of 320×240 pixels.

In terms of computation, each iteration requires the computation of 3×3 gradients in each (non-discarded) pixel which is very fast. The most costly step is to fill the $N \times 6$ interaction matrix $\mathbf{L}_\mathbf{Z}$ to compute the control law. Note that $\mathbf{L}_\mathbf{Z}^\top\mathbf{L}_\mathbf{Z}$ used to compute the pseudo-inverse is a 6×6 matrix which is very fast to invert. Without any specific optimization, the code runs in about 60 ms per frame on a standard laptop. The method is suitable for real-time experiments.

1) Using a synthetic depth image to define the desired position: In the first experiment, we consider the case where the desired position is defined in a simulation environment using a model of the scene. This kind of approach can typically be beneficial in applications where a 3D model is known but one wants to define different tasks without the need for depth maps acquired in situ. In that case we render the desired depth image from a 3D model of the scene instead of using one acquired from the sensor.

For this experiment, we built a 3D model of the scene using the ReconstructMe[3] software. We then defined the 3D pose we wanted the robot to reach, with respect to this model. The desired depth map corresponding to this pose was rendered using Ogre3D[4] using the actual depth camera calibration parameters. In this case the depth sensor thus needs to be calibrated.

[2]http://openkinect.org/
[3]http://reconstructme.net
[4]http://www.ogre3d.org/

Figure 6 (1-b) shows an example of such a rendered image for the model of Figure 5. The white points in this image correspond to unmodeled areas, for which the rendering gives an infinite depth value. They are treated as unavailable data, as for Kinect depth maps, and are excluded from the feature set (see Section III).

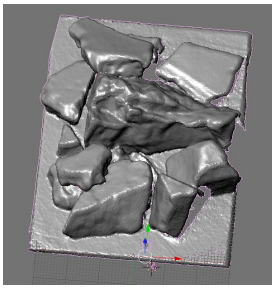


Fig. 5. 3D model used for generating the desired depth map in Figure 6.

The positioning task consists for the robot in minimizing the error between this rendered depth map and the current one acquired by the Kinect sensor.

The results of this experiment are shown in the Figures 6 and 7. The first image (1-a) shows the RGB view provided by the Kinect for the initial position. This image is never used in the control scheme and is only given here for a better understanding of the setup. The depth maps are shown in the second row, and the last row gives the corresponding error, i.e. the difference between the desired and the current depth maps, unavailable data being discarded as explained in III-A. The difference images are scaled so that a plain grey frame (3-b) corresponds to a null error, and thus to the good achievement of the task. Figure 6 (3-a) gives a vizualization of the error in the initial position.

The corresponding quantitative values for the task error, the 3D positioning errors and the velocities are given in Figure 7. Figure 7 (b) shows the repartition of the depth errors in the initial frame of figure 6 (3-a). The maximum error is about 50 cm in depth. The peak at 0 corresponds mostly to white pixels of Figure 6 (1-b) that are not considered.

Note that one iteration corresponds to one execution of the control loop, that is the computation of (13) for the current image and the transmission of this velocity command to the robot. Figures (c) and (d) show that with an initial error of 15cm in translation and 10° in rotation, the positioning task is properly performed, as indicated by the low residuals. Here the final accuracy depends on the quality of the model. To evaluate the final accuracy of the system itself we thus consider in the next experiment a desired depth map acquired at the desired position.

2) Robustness to occlusions and scene changes: In the second experiment, we evaluate the robustness of our approach with respect to partial occlusions or modifications of the observed scene. First, the desired depth map is acquired at the desired position, then the robot is moved to the initial state in which the servoing is launched. The goal here is to regulate the error between this reference depth map and the current depth map. The initial scene is illustrated in Figure 9 (1-a). During the task achievement, someone entered the sensor field, removed an object and put it back several times. Some selected frames of this sequence are shown in Figure 9. The full video of this sequence is provided as supplementary material. The initial and final positions are illustrated in the first and last columns, while columns (b) and (c) show examples of occlusions. Note that at the end of the sequence the white bear has been completely removed

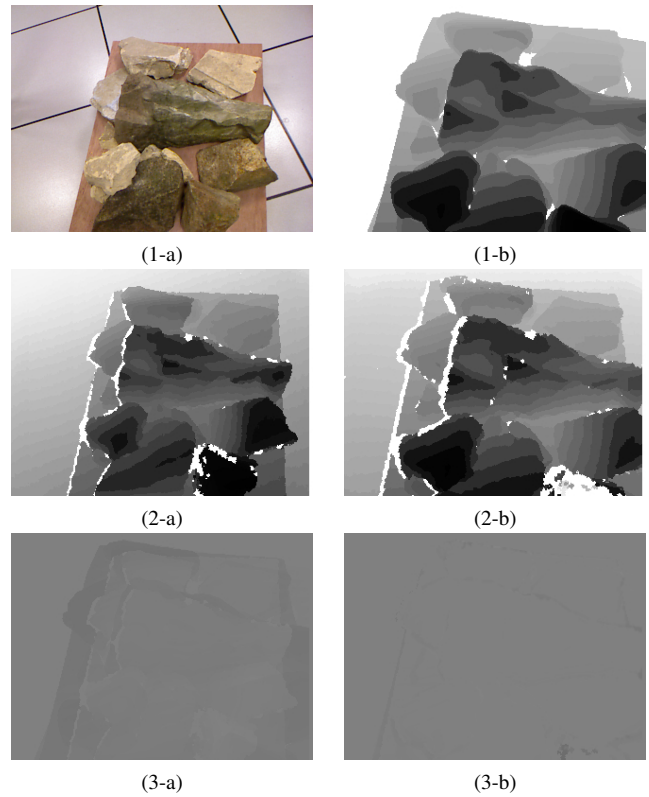


Fig. 6. First experiment. First column corresponds to the initial position. The RGB view from the Kinect (1-a) is not used in the algorithm. (2-a) Initial depth map, where white parts correspond to unavailable data. (3-a) Difference between the initial and desired depth maps. Second column corresponds to the end of the motion. The desired depth map (1-b) was rendered using a 3D model of the scene. (2-b) shows the final depth map, which minimizes the difference (3-b) with the desired one (1-b).

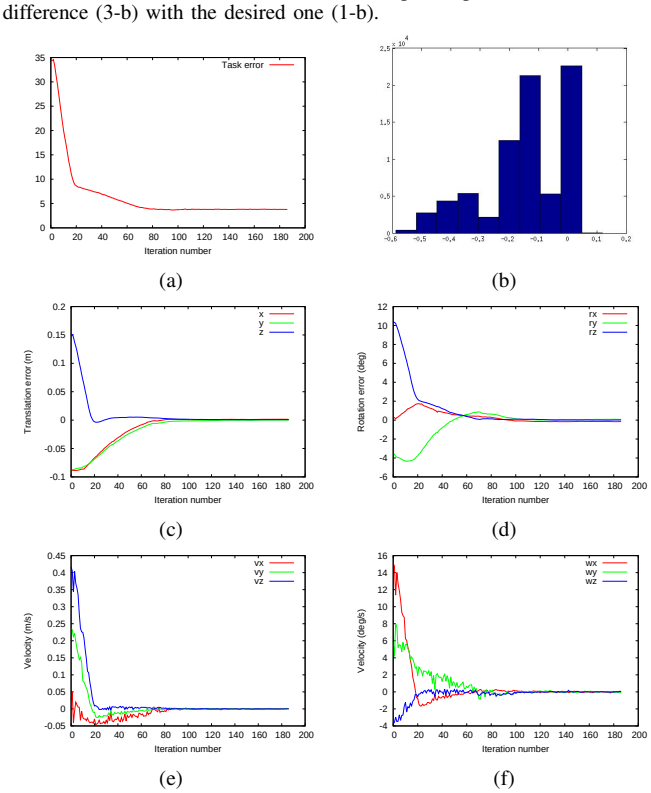


Fig. 7. First experiment. (a) Task error, (b) histogram of depth errors (m) corresponding to the initial frame, (c) translational part of positioning error, (d) rotational part of positioning error, (e) translational velocities, (f) rotational velocities.

from the scene, and the final depth map (Figure 9 (2-d)) is thus different from the desired one (Figure 8 (b)). This difference appears in the final difference image (Figure 9 (4-d)) and the task error function (Figure 8 (a)). However, despite the scene modifications and occlusions, the positioning task is successfully achieved, as shown by the convergence of the positioning errors in Figure 8 (c). The robustness of our control scheme to perturbations is the result of the use of M-estimation (see III-B). The effect of M-estimation is illustrated on the third row of Figure 9, where the relative weights of each data in equation (12) are represented. Black pixels correspond to rejected values and brightest ones to inliers. Figure 9 (3-b), (3-c), and (3-d) show that the perturbations are correctly detected since the corresponding pixels are given a smaller weight.

Figures 8 (c) show that with an initial error of 17cm in translation and 20° in rotation, the positioning task is properly achieved with a remaining error of less than 3mm in translation and 0.4° in rotation. Given the low depth resolution of the sensor and its noisy measurements, this corresponds to a good achievement of the task.

Finally, note that in this scene the smoothness assumption was not verified everywhere since large depth discontinuities exist at the border of the objects, for example between the table and the floor. This experiment thus shows that the method is successful beyond its initial assumption. This is due to the fact that the points corresponding to discontinuities are a minority and are detected as outliers by the M-estimator.

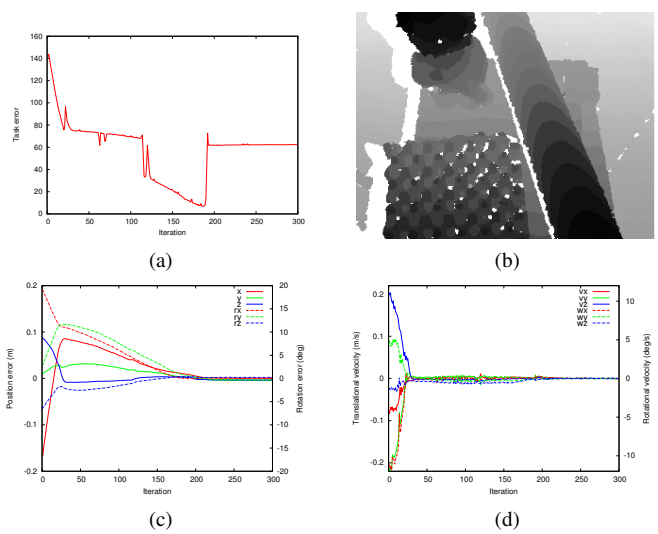


Fig. 8. Second experiment. (a) Task error, (b) desired depth map, (c) positioning errors, (d) velocities.

3) Experimental analysis of the convergence domain: We want to underline that the above experiments are examples from a large number of experiments that we performed using different initial poses and scenes. As for IBVS approaches, only the local asymptotic stability can be demonstrated (see [4]). Thus we can only assess the performances in terms of convergence domain from experiments. In order to empirically evaluate the convergence domain we run multiple positioning tests from different initial positions and recorded the convergence succes and failure. One test is deemed to have converged if both the norm of the transation error vector and the norm of the rotation error vector get smaller than 1 cm and 1° respectively, in less than 500 iterations. Those tests were performed using the 3D model of Figure 5 in simulation, without adding occlusions. Simulation allows us to handle exhaustive testing with hundreds of different initial positions. The initial poses are chosen so that the camera center is placed on a regular 3D grid centered on the desired pose in x and y, and with z varying from 0.3 m to 4.7 m where the desired z camera position corresponds to 1.5 m. The orientation is set so that the desired and initial depth maps overlap. This setting leads to large variations of x-axis and y-axis rotations, from -60° to 60°. We also considered rotation around z-axis, by running one full set of simulations with 0° z-axis rotation and another one with 30° z-axis rotation. Figure 10 shows the resulting convergence domains, which as can be seen are considerable. Note that joint limits are not considered in this simulation test. From Figure 10 we can see that the convergence domain has an approximate radius of more than 1 meter along the x and y directions in this setting, and even more in the z axis where convergence can be obtained from more than 2 m above the desired position. The methods handles indeed easily a large initial error on the depth axis which generates a large velocity component on this axis to compensate for the important depth map error. Note that the convergence domain is scene-dependant, and large structural elements with smoothly varying depth (planes or large rocks in this example) will generally lead to a larger convergence domain than scenes with high frequency depth variations.

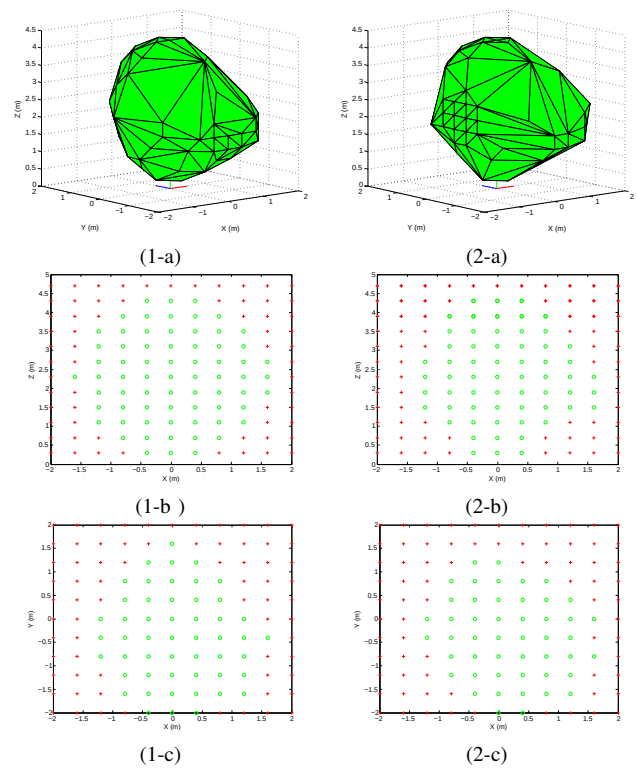


Fig. 10. Experimental evaluation of the convergence domain. Result are given for zero z-axis rotation in column (1) and 30° z-axis rotation in column (2). The first row shows for both cases the convex hull of the points from where the system successfully converged. Rows (b) and (c) shows slices at the desired y and z planes respectively. The poses that converged are shown by green circles and those that diverged by red crosses.

B. Non honolomic robot navigation using depth map memory

In the experiments above, we considered positioning tasks on a 6 dof robot, in various conditions. In this part, we propose to apply our depth-based approach to a navigation task on a wheeled robot (Pioneer P3-DX). We assume here that we have stored a sequence of depth maps acquired during a manual navigation stage. These depth maps can be considered as a sensory memory that is then used for

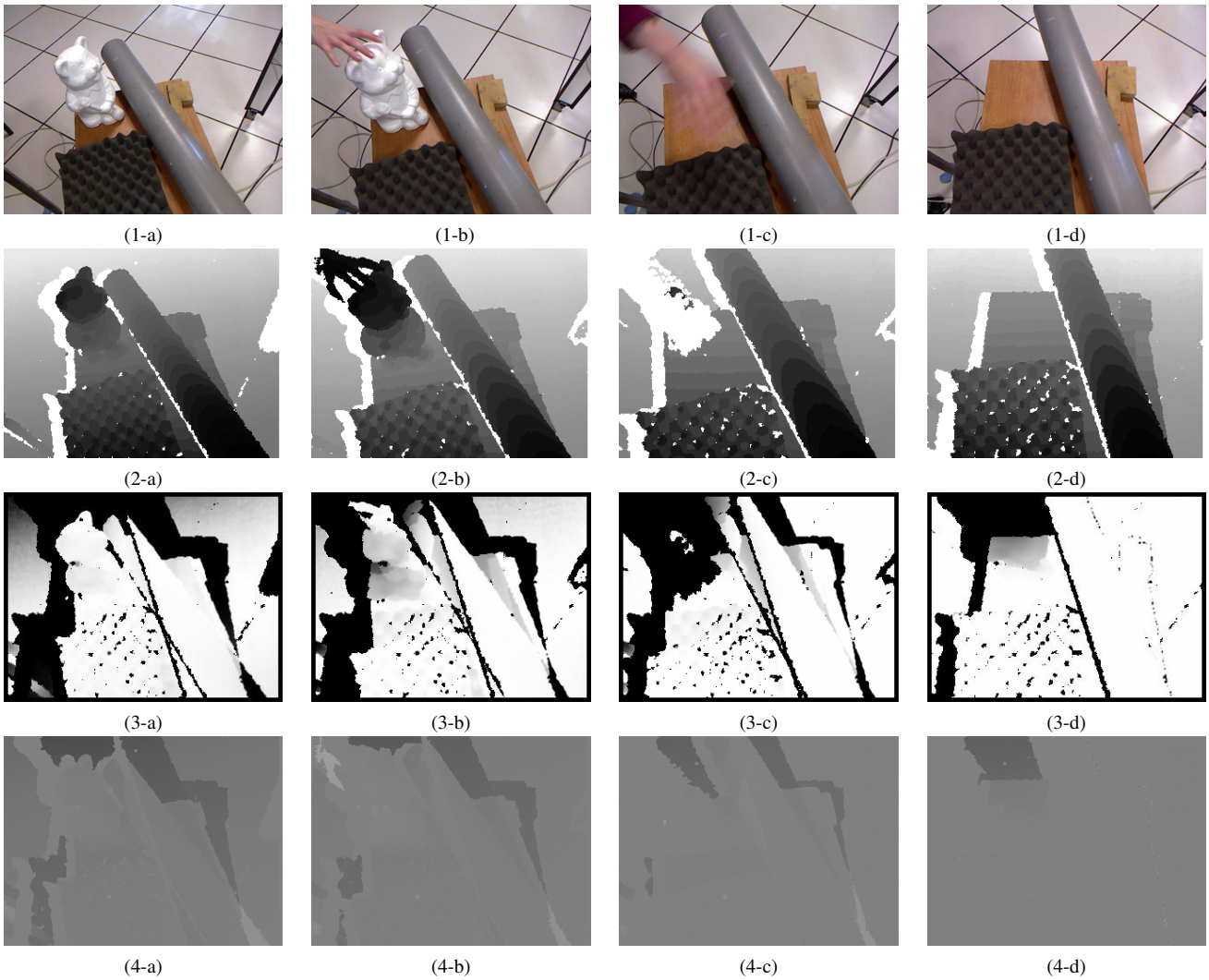


Fig. 9. Selected frames of the second experiment. Columns correspond to frames 1, 15, 69 and final frame respectively. Those frames illustrate occlusions and object removal (1-b) (1-c) (1-d). The first row gives the RGB view from the Kinect, which is not used in the algorithm but shows the setup. The depth maps are represented in the second row. The images of the third row represent the weights of each pixel in the M-estimation. Black pixels are discarded. Frames (3-b) (3-c) (3-d) show that occluded areas are given a very low weight. Fourth row: difference between the initial and desired depth maps.

the autonomous navigation. The navigation task is thus defined as a succession of positioning tasks using the successive stored data as desired depth maps (see Figure 11). In this case, each positioning sub-task is performed in a similar manner as in the previous experiments, but controlling 2 dof only: the forward translational motion and the in-plane rotation. The switch from one reference depth map to the next is based on a simple threshold on the error decrease.

Figure 12 shows some samples from our navigation sequence. The first row represents the robot during the learning phase, that is when it was manually controlled and acquiring the reference depth maps. The second row gives the autonomous navigation results. Although no ground truth measurement was available for this task, this figure shows that the robot closely follows the path of the learning phase, using the stored maps as references.

Figure 13 provides typical depth maps from this navigation sequence, along with the corresponding errors when the reference frame is changed. Note that since the robot is non-holonomic, one cannot ensure its convergence to the 3D position corresponding to the desired depth map, which explains that the error images are not as good as for the previous 6 dof positioning tasks. Note also that one possible indetermination can occurr in the case of a long corridor with no

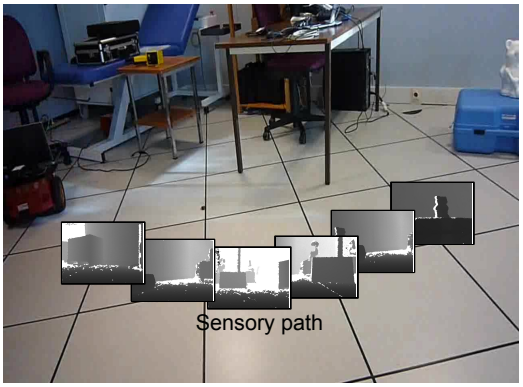


Fig. 11. A sequence of depth maps has been acquired in a manual navigation step. The navigation task is then defined using each depth map from the memory as an intermediate desired depth map for the controller.

door to mark a depth variation, since several different poses would lead to the same depth map (the interaction matrix would then be rank deficient). This however would not be an issue for tasks such as moving forward while being centered in a corridor. This experiment shows the feasibility of the approach for indoor navigation.

V. Conclusions

We have demonstrated that it is possible to use a dense depth map directly to control robot motion. The goal position can be defined by a single depth map either directly acquired from that position or synthetically rendered. The main advantage of our approach is that it does not require any pose estimation, feature extraction or matching step. Moreover, when the depth map is obtained from an active sensor, the resulting approach is not sensitive to illumination changes as photometric approaches can be. Some limitations can appear with the use of active sensors such as Kinect RGB-D camera, in particular the noise and the absence of some measurements. We show however that those issues can be overcome thanks to the use of M-estimators and basic image pre-processing.

Acknowledgment

This work was supported by ReV-TV FUI. The authors would like to thank François Chaumette for helpful discussions.

References

[1] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *Int. Journal of Computer Vision*, 56(3):221–255, February 2004.

[2] P.J. Besl and H.D. McKay. A method for registration of 3-D shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[3] F. Chaumette and S. Hutchinson. Visual servo control, Part I: Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90, December 2006.

[4] F. Chaumette and S. Hutchinson. Visual servoing and visual tracking. In B. Siciliano and O. Khatib, editors, *Handbook of Robotics*, chapter 24, pages 563–583. Springer, 2008.

[5] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *IEEE Int. Conf. on Robotics and Automation*, pages 2724–2729, 1991.

[6] C. Collewet and E. Marchand. Photometric visual servoing. *IEEE Trans. on Robotics*, (99):1–7, 2011.

[7] C. Collewet, E. Marchand, and F. Chaumette. Visual servoing set free from image processing. *IEEE Int. Conf. on Robotics and Automation*, pages 81–86, May 2008.

[8] A.I. Comport, E. Marchand, and F. Chaumette. Statistically robust 2-D visual servoing. *IEEE Transactions on Robotics*, 22(2):415–420, 2006.

[9] J. Corso, D. Burschka, and G. Hager. Direct plane tracking in stereo images for mobile navigation. In *IEEE Int. Conf. on Robotics and Automation (ICRA '03).*, volume 1, pages 875–880, Sept 2003.

[10] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli. Depth mapping using projected patterns, May 2010. Patent US 20100118123.

[11] H. Gharavi and S. Gao. 3-D Motion Estimation Using Range Data. *IEEE Trans. on Intelligent Transportation Systems*, 8(1):133–143, March 2007.

[12] G.D. Hager and P.N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, Oct 1998.

[13] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D Mapping: Using depth cameras for dense 3D modeling of indoor environments. In *Int. Symposium on Experimental Robotics (ISER)*, 2010.

[14] B.K.P. Horn and J.G. Harris. Rigid body motion from range image sequences. *CVGIP: Image Understanding*, 53(1):1–13, January 1991.

[15] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, August 1981.

[16] P.-J. Huber. *Robust Statistics*. Wiler, New York, 1981.

[17] V Lepetit and P Fua. Monocular Model-Based 3D Tracking of Rigid Objects: A Survey. In *Foundations and Trends in Computer Graphics and Vision*, pages 1–89, 2005.

[18] A. Makadia, A. Patterson, and K. Daniilidis. Fully automatic registration of 3d point clouds. In *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 1297–1304, 2006.

[19] E. Marchand, F. Spindler, and F. Chaumette. ViSP for visual servoing: A generic software platform with a wide class of robot control skills. *IEEE Robotics and Automation Magazine*, 12(4), December 2005.

[20] P. Martinet, J. Gallice, and D. Khadraoui. Vision based control law using 3D visual features. In *World Automation Congress, Robotics and Manufacturing systems*, volume 96, pages 497–502, 1996.

[21] S. May, D. Droeschel, D. Holz, S. Fuchs, E. Malis, A. Nuchter, and J. Hertzberg. Three-dimensional mapping with time-of-flight cameras. *Journal of Field Robotics*, 26(11-12):934–965, 2009.

[22] RA. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A.J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *Int. Symposium on Mixed and Augmented Reality*, 2011.

[23] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *Int. Conf. on 3-D Digital Imaging and Modeling*, pages 145–152, 2001.

[24] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. Journal of Computer Vision*, (47):7–42, 2002.

[25] G. Silveira and E. Malis. Real-time visual tracking under arbitrary illumination changes. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–6, June 2007.

[26] G. Silveira and E. Malis. Unified direct visual tracking of rigid and deformable surfaces under generic illumination changes in grayscale and color images. *Int. Journal of Computer Vision*, 89(1):84–105, 2010.

[27] A. Swadzba, B. Liu, and J. Penne. A comprehensive system for 3D modeling from range images acquired from a 3D ToF sensor. In *Int Conf. on Computer Vision Systems (ICVS)*, 2007.

[28] C Teulière and E Marchand. Direct 3D servoing using dense depth maps. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'12*, pages 1741–1746, Vilamoura, Spain, October 2012.

[29] WJ. Wilson and CC W. Hulls. Relative End-Effector Control Using Cartesian Position Based Visual Servoing. *IEEE Trans. on Robotics and Automation*, 12(5), 1996.

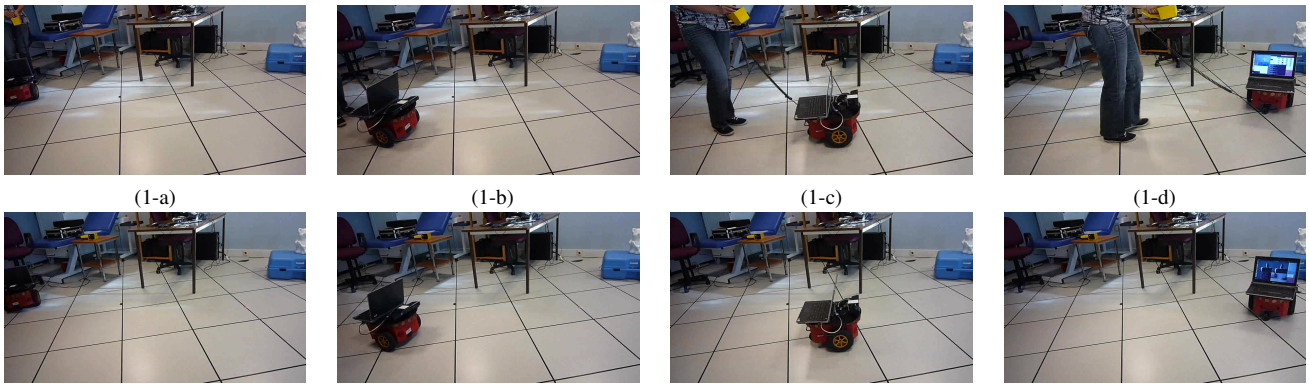
[30] M. Yamamoto, P. Boulanger, J-A. Beraldin, and M. Rioux. Direct estimation of range flow on deformable shape from a video rate range camera. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(1):82–89, 1993.

Appendix

Depth information has already been used in position-based visual servoing. For example, [20] proposed to use the 3D coordinates (X, Y, Z) of a set of 3D points as features to be regulated in a proportional control law. In other words, the positioning task was expressed as the regulation of the feature $\mathbf{P} = (X_1, Y_1, Z_1, ..., X_N, Y_N, Z_N)$ to a reference feature $\mathbf{P}^* = (X_1^*, Y_1^*, Z_1^*, ..., X_N^*, Y_N^*, Z_N^*)$ corresponding to the 3D coordinates of the set of points at the desired robot position. The interaction matrix related to a single 3D point is then given by [3][20]:

$$\mathbf{L}_P = \begin{bmatrix} -1 & 0 & 0 & 0 & -Z & Y \\ 0 & -1 & 0 & Z & 0 & -X \\ 0 & 0 & -1 & -Y & X & 0 \end{bmatrix}. \tag{14}$$

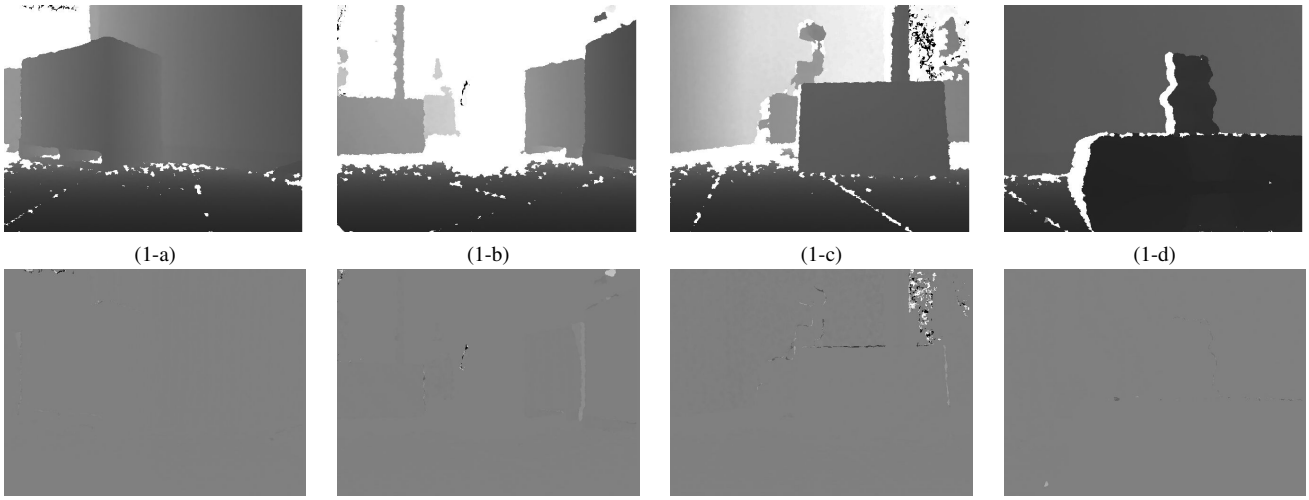
At first sight, the depth components of this kind of 3D feature $(X_1, Y_1, Z_1, ..., X_N, Y_N, Z_N)$ could seem very close to the vector formulation $\mathbf{Z} = (Z_1, ..., Z_N)$ that we defined in Section II-C. However, a key difference with respect to our approach is that [20] uses a sparse set of 3D features. Consequently, in [20] a matching step is required to determine the feature values through the sequence, and the range flow equation (3), based on a smoothness assumption, does not hold in the sparse case. On the contrary, one of the key advantages of the method we propose, is that it does not require any feature extraction nor matching step and uses directly the dense depth information from the range sensor thanks to the range flow equation. That is why the interaction matrix related to the depth map is given by (9) while the interaction matrix related to the depth of point is given by the last row of (14).



(1-a) (1-b) (1-c) (1-d)

(2-a) (2-b) (2-c) (2-d)

Fig. 12. Extracted frames from the navigation experiment. The first row corresponds to the manual navigation step where the depth maps are memorised. The second row shows the autonomous navigation using this sensory memory.



(1-a) (1-b) (1-c) (1-d)

(2-a) (2-b) (2-c) (2-d)

Fig. 13. Extracted frames from the navigation experiment. The first row shows the depth map observed before switching to a new reference frame. The error images are shown on the 2rd row.