Real-time Model Based Visual Servoing Tasks on a Humanoid Robot

Amine Abou Moughlbay, Enric Cervera and Philippe Martinet *

Abstract Several model based techniques have been used to apply various domestic service tasks on humanoid robots (through teleoperation, learning, ...). But for many reasons, it is more suitable to study the interaction between the robot and its environment using the Sensor Based Control in these cases. In this paper we present a work of integration of real-time visual servoing techniques in performing self localization and different manipulation tasks on a humanoid robot in closed loop.

Real-time model based tracking techniques are used to apply 3D visual servoing tasks on the Nao humanoid robot. Elementary tasks used by the robot to perform a concrete scenario are detailed with their corresponding control laws. Experimental results are presented for the following tasks: self-localization of the robot while walking, head servoing for the visibility task, detection, tracking and manipulation of environment's objects.

Amine Abou Moughlbay
Institut de Recherche en Communications et Cybernétique de Nantes, Ecole Centrale de Nantes, 1 rue de la noë, 44321 Nantes - France, e-mail: `Amine.Abou-Moughlbay@ec-nantes.fr`

Enric Cervera
Robotic Intelligence Lab, Jaume-I University, 12071 Castello - Spain,
e-mail: `Ecervera@icc.uji.es`

Philippe Martinet
Institut de Recherche en Communications et Cybernétique de Nantes, Ecole Centrale de Nantes and Institut Pascal in Clermont Ferrand - France,
e-mail: `Philippe.Martinet@irccyn.ec-nantes.fr`

* This work has been funded by the French National Agency of Research under the reference ANR-10-SEGI-002.

1 Introduction

To perform manipulation tasks, the robot interacts with its environment through contact points, such as applying forces and moments on objects. By controlling the position and velocity of these points, as well as the forces acting on them, the robot performs the desired tasks. In complicated and uncertain environments, end-effector motion may be subject to online modifications in order to accommodate unexpected events or to respond to sensor inputs. When controlling the interaction between manipulator and environment, the desired task is often specified in the Operational Space and requires precise control of the end-effector motion. Joint Space Control schemes are not suitable in these situations. Thus programming of service and manipulation tasks is most conveniently accomplished by directly specifying data at the contact points, rather than specifying the joint positions and velocities required to achieve them [1].

To apply service tasks, many approaches have been used: different works have been carried out in the area of teleoperation, by controlling a robotic system to perform tasks at a distance using a multi-modal human-system interface which provides sensory feedback to the operator and allows him to interact with the remote environment by mapping his actions. It was applied to bi-manual manipulation and walking [2], haptic interface for mobile teleoperators [3] and recently on a teleoperation system with haptic device for micro-manipulation [4].

Furthermore, many practical learning control systems are used to control complex robots involving multiple feedback sensors and multiple command variables during both repetitive and nonrepetitive operations [5]. The issue of teaching a robot to manipulate everyday objects through human demonstration has been studied by [6] who proposed a method that enables a robot to decompose a demonstrated task into sequential manipulation primitives, series of sequential rotations and translations [7]. Other earlier works used also the Model Based Control strategies, executed by automatically generating a control sequence that moves the robot to the states specified by the program to develop executives that emphasize model based approaches and deep integration of automated planning [8].

Self-localization is one of the fundamental problems in robotics, as in many applications a robot needs to know its location in order to perform its tasks. Most of the indoor localization algorithms use particle based filters or Kalman type filters to solve the problem of noisy sensors and controls. Particle filters inherently help solve the problem of ambiguous landmarks, whereas Kalman filters must track multiple hypotheses to work in ambiguous environments [9]. In some divisions of RoboCup, algorithms are very well established, given the rich sensor data provided by laser scanners, omni-directional cameras etc. However, in more general cases, there are substantial sensor limitations particularly with the rapid motion of the camera, and the need for active perception [10]. In addition to odometry sensors, some recent alternatives use also the ambient magnetic field to control the heading of a robot in case of one-dimensional localization problem [11].

As previously presented, most actuated systems use sensors to obtain information about their environment. These can be a camera, ranging devices, or temperature and force sensors. Among all these feedbacks, the visual information provides the most important and instant cues for perception of the interaction with the working environment. Compared to already mentioned methods, visual servoing provides very efficient solutions to control robot motions. It supplies high positioning accuracy, good robustness to sensor noise and calibration uncertainties, and reactivity to environment changes [12].

In this paper, only Sensor Based Control formalism is used to perform the desired tasks. More especially, 3D visual feedback data and Model Based Tracking (MBT) techniques are used to execute, in real-time and closed loop, many tasks on the humanoid mobile robot Nao in a semi structured environment. The robot's camera is calibrated, and a rough geometric model of the objects is available (doors, tables, pieces to grasp ...). The envisioned scenario consists of a Nao robot which can carry out service tasks, moving around the room and manipulating objects. One of the missions that can be requested from such robot would be to "Pick up the orange and green pieces and deposit them on the table nearby the door" (see Fig. 1). But to execute this mission, many problems should be addressed and resolved:

- How can the robot localize itself wrt. its environment?
- How to decompose the robot's mission into elementary tasks?
- What information should be given to the robot to execute each task?
- What are the more useful techniques the robot can use to perform these tasks?

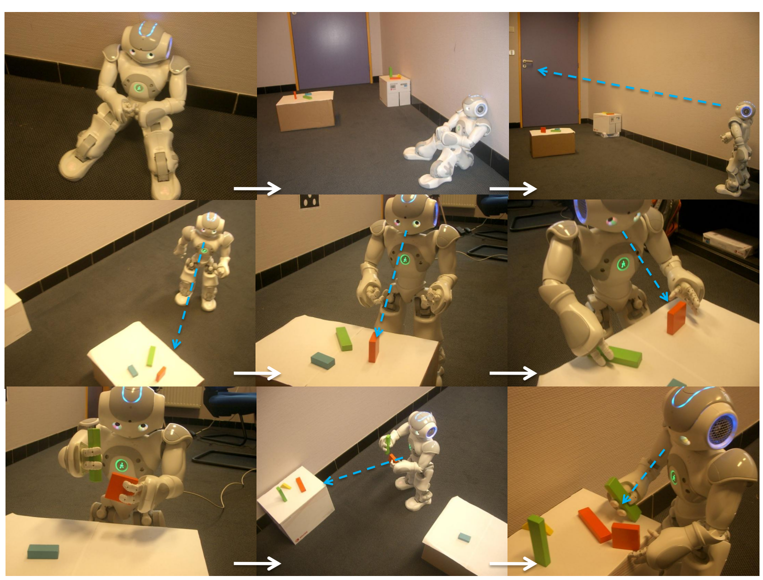


Fig. 1 Nao picks up the orange and green pieces and deposits them on the table near the door

In the second section of this paper, we present the architecture of the Nao robot and the Model Based Tracking technique. In section 3, we define the common elementary tasks and frames employed during the application of the desired tasks. Implemented tasks are presented in section 4 and the experimental results in section 5. The first part of this latter includes the localization of the robot in its environment when walking. The second one includes the object tracking and manipulation by visual servoing. The final section draws some conclusions and outlines future works.

2 System Architecture

2.1 Nao Architecture

Nao Robot [13], developed by Aldebaran robotics, is a biped robot with 25 Degrees of Freedom (DOF). It has 3-fingered robotic hands used for grasping and holding small objects (it can carry up to 300g using both hands). It is equipped with: 2 ultrasound devices situated in the chest that provide space information in a range of 1 meter, 2 cameras situated on the top and bottom of the head, 2 bumpers (contact sensors on the robot's feet), a gyrometer and an accelerometer (to determine whether the robot is in a stable or unstable position).

2.2 Visual servoing and tracking techniques

A large variety of positioning or target tracking tasks can be implemented by controlling from one to all DOF of the system. For whatever the sensor configuration, which can vary from one camera mounted on the robot end-effector to several free-standing cameras, a set of visual features $\mathbf{s}$ has to be designed from the visual measurements obtained from the system configuration $\mathbf{x}(t)$ $(\mathbf{s} = \mathbf{s}\,[\mathbf{x}(t)])$, allowing control of the desired DOF. A control law is thus designed so that these features $\mathbf{s}$ reach a desired value $\mathbf{s}^*$, defining a correct realization of the task. Indeed, if the camera velocity is considered as input of the robot controller, the control law which performs the desired exponential decoupled decrease of the error $\mathbf{e} = (\mathbf{s} - \mathbf{s}^*)$ is:

$$\mathbf{V}_c = -\lambda \mathbf{L}_s^+ (\mathbf{s} - \mathbf{s}^*) \tag{1}$$

where λ is a proportional gain that has to be tuned to minimize the time-to-convergence, and $\mathbf{L}_s^+$ is the pseudo-inverse of the interaction matrix (for more details refer to [14]).

Many tracking tools have been implemented in several visual servoing toolboxes [15]. On the Visual Servoing Platform (ViSP) [16], we find a dot tracker, a moving

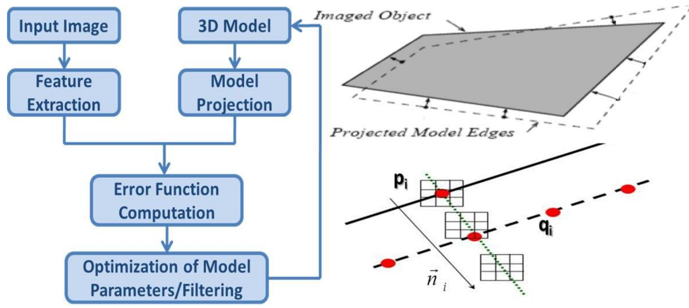


Fig. 2 Model Based tracking system (left) using the Moving edge detection technique (right)

edges tracker, and a 3D model based tracker. The last one [17] tracks a 3D model thanks to the moving edges method, using a virtual visual servoing technique. It requires a 3D model and needs to compute the initial pose which is used to project the model on the image. The tracking method assumes that the pose corresponding to the previous image is known, the new lines are tracked, and the goal is to move the pose to match the object in the new image with the projection of the model.

The error function (err) between image features p_i and model projection q_i is thus minimized along the normal direction **n** (see Fig. 2):

$$err = \sum_i \Delta(p_i, q_i) = \sum_i |(q_i - p_i) \cdot (n_i)| \tag{2}$$

3 Scenario and Task Identification

3.1 Frames and Control Law

To execute the different tasks, we should define frames on the robot's body and environment's items (see Fig. 3). Using the inverse kinematic model of the robot and the 3D visual servoing technique, the general control law used to define a task is given by:

$$\dot{\mathbf{q}} = -\lambda (\mathbf{L_s J})^+ (\mathbf{s} - \mathbf{s}^*) \tag{3}$$

where $\dot{\mathbf{q}}$ is the robot's joint velocity and $\mathbf{J}$ is the kinematic matrix which relates camera velocity with the robot's joint velocity ($\mathbf{V}_c = \mathbf{J}\,\dot{\mathbf{q}}$).

3.2 Generic Tasks

While Nao is a capable platform, due to the complexity of the problem to be solved, the capabilities of the robot can be enhanced and the complexity of the problem can

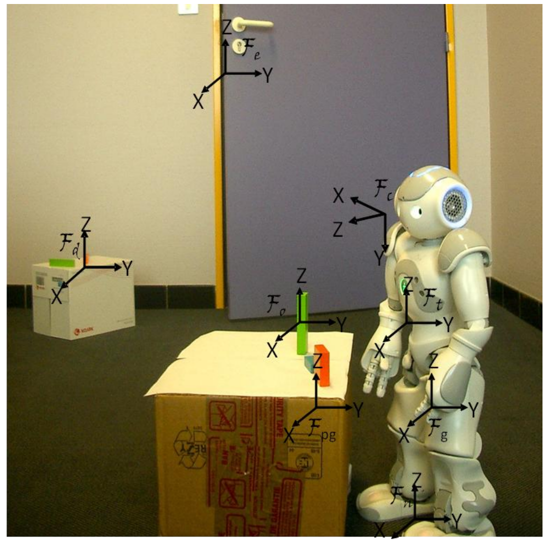


Fig. 3 Useful frames in Nao's environment:

In Nao's body we consider the following frames: Nao's space frame $\mathscr{F}_n$ (between robot's feet), $\mathscr{F}_t$ on robot's torso, $\mathscr{F}_c$ a camera attached frame, robot's hand frame $\mathscr{F}_h$, robot's gripper frame $\mathscr{F}_g$ and a pre-grasping frame $\mathscr{F}_{pg}$.
The following frames are defined in the robot's environment: object's frame $\mathscr{F}_o$, desired object's pose frame $\mathscr{F}_d$ and environment's frame $\mathscr{F}_e$ for the localization task.

be reduced by decomposing it into simpler generic tasks. Thus, the desired scenario is executed by answering the following questions:

1. **Where am I ? → Self-Localization Task**
   Depending on the desired mission to be executed, the robot searches for the corresponding set of models in its environment to be localized wrt. them (table, door, light switch, corner, ...). After detecting the items in the field of view of the robot's camera, and choosing the nearest one to the object to manipulate ($\mathscr{F}_e$), the robot calculates, using the MBT technique, its position/orientation wrt. the item in form of a homogeneous transformation matrix (${}^n\mathbf{M}_e$).

2. **What must I do and How to do it ? → Task Scheduler**
   The desired scenario is interpreted to the robot's language and decomposed into elementary tasks [18]:
   - Localize the desired objects to manipulate
   - Move in the appropriated direction
   - Detect and track the desired objects
   - Keep these pieces in the robot's field of view
   - Move the robot's arms and grasp these pieces
   - Go to the desired table and deposit objects

3. **Which data will I use ? → Definition of used data**
   To execute the defined elementary tasks, the scheduler chooses the appropriate models for each one: models of the objects to manipulate, model of the environment's items (for the self-localization task), data of the robot's camera, parameters of the robot's gripper and the used items...

4. **Where to move ? → Robot Locomotion Task**
   This task is used to move to the region where the task is to be executed. Knowing the position of at least one item of the environment (${}^{n}\mathbf{M}_{e}$), the robot walks in the appropriate direction of ($\mathscr{F}_e$) until enterring the range of a defined distance from the object. During this task, an obstacle avoidance technique is used to adapt the robot's trajectory [19].

5. **How to perceive ? → Detection and Tracking Tasks**
   Using the MBT technique of ViSP, the tracker is manually initialized and the pose of the desired item is determined. Hence it allows us to track in real-time the pose of the object to manipulate. An automatic re-initialization of the tracker is implemented: it uses the last poses of the tracked object to be reinitialized in case of a failure due to an occlusion or the fast motions of the robot's camera (especially when walking). The automatic transition between the tracking of different objects is also implemented using the knowledge of the rough relative position between environment's objects.

6. **How to keep the concerned points in the robot's field of view ? → Visibility Task**
   This task consists of controlling the robot's head position/ orientation to focus a (fixed/mobile) point of the environment (item's center, gripper, virtual point...) in the center of the camera's image. This task can be used in hand-eye coordination for dynamic grasping of objects by focusing on the gripper-item midpoint, to keep the robot's hand and object in the robot's field of view. The used DOF in this task depends on the geometry of the robot's head and the desired complexity of the task. The head's Yaw/Pitch can be controlled to focus on the object's center, and a more complex task can also control the distance between the robot's head and the object.

7. **How to perform the manipulation ? → Grasping Task**
   This task uses the hand's control point and allows the robot to move it to a desired static/mobile pose. This task can be used to perform pre-grasping, grasping, and displacing objects tasks. In case of pre-grasping task, the goal position (${}^{g}\mathbf{M}_{pg}$) is determined using the grasping strategies. These strategies depend on the geometry of the object to manipulate and the geometry of the robot's gripper. The grasping strategy controls the relative position and/or the angle between the gripper and the item to grasp. For a grasping task, the same technique is considered: the robot's arm moves to a desired pose by minimizing the relative distance/orientation between the object and the robot's hand (${}^{g}\mathbf{M}_{o}$) in the Gripper's Frame [20].

4 Tasks Definition

In this part, we present preliminary results of the general scenario detailed in the previous section applied on the humanoid robot Nao. The experimented tasks are: the self-localization of the robot when walking, and the head and arm's visual servoing to track and grasp an object. During the execution of the first task, only a rough model of the door and a part of the room is used, and for the second one only the approximate model of the item to grasp is given. Note that no other exteroceptive data is given from the robot's environment.

4.1 Self-Localization Task during Locomotion

During this task, the robot tracks the door and thus it is localized wrt. the environment while walking. We use the model of the door and the lines of the room around it to initialize the MBT which gives then the pose of the door in the camera's frame $({}^{c}\mathbf{M}_{e})$. From the given pose, the robot's pose can be calculated: ${}^{e}\mathbf{M}_{n} = ({}^{c}\mathbf{M}_{e})^{-1}\,({}^{n}\mathbf{M}_{c})^{-1}$.

Note that the tracker was automatically reinitialized each time the tracking failed due to large camera displacements during walking; it used the last found pose of the object to reinitialize the tracking. In our experiments, the robot walks in open loop in the direction of the door for a distance of 1 meter.

4.2 Detection and Tracking Task

The MBT is used to track simple item models; it's initialized manually at the beginning of the application. Afterwards, the model is automatically detected and tracked; this tool allows us to determine instantly the pose of the desired item frame in the robot camera's frame $({}^{c}\mathbf{M}_{o})$.

4.3 Visibility Task

Throughout this application, the visibility task is used for controlling the robot's head orientation to focus the item's center in the center of the camera's image. Two DOF are used by this task to control the head's Yaw and Pitch. The task's goal is thus to regulate exponentially ($\dot{\mathbf{e}} = -\lambda\,\mathbf{e}$) the horizontal and vertical position of the center of the object projection $\mathbf{s}_{x,y} = {}^{c}\mathbf{T}_{o(x,y)}$ to zero ($\mathbf{s}^{*} = (0,0)$).

Using the object 3D pose ${}^{c}\mathbf{T}_{o} = (X,Y,Z)^{T}$, and the 2D pose $(x,y)^{T}$ of the tracked point (projection of 3D point in the normal image plane), we apply the control law defined in (3) using the visual primitive $\mathbf{s} = (x, y)$ and its corresponding interaction matrix $\mathbf{L_s}$ given by:

$$\mathbf{L_s} = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+y^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+x^2 & -xy & -x \end{bmatrix} \tag{4}$$

Note that for this task the $\mathbf{J}$ matrix in (3) uses the Jacobian of the robot's head control point calculated from the robot's geometric model.

4.4 Pre-Grasping Task

Through this task the robot's arm is supposed to move close to the item to grasp. According to Nao's gripper's geometry (of one DOF) and the item's shape (rectangular model), the number of constrained DOF and the pre-grasping position is predefined in the robot's manipulation parameters. This position is defined wrt. the object's position in Nao's frame $^{n}\,(^{o}\mathbf{M}_{pg})$.

In our case, regarding the gripper's and item's shapes, 4 DOF are enough to execute this task: 3 DOF constraints the gripper's pose and 1 DOF (Yaw angle) for the gripper's orientation. Furthermore, the pre-grasping distance is fixed to 5 cm, but we should not forget that the tracker gives the item's pose in real-time, thus the desired pre-grasping pose is calculated in closed loop. The task's target is then to move the robot's arm to the pre-grasping pose. The task's error is extracted from the relative pose between the gripper and pre-grasping point $(^{g}\mathbf{M}_{pg})$ which is regulated to zero.

Note that $^{g}\mathbf{M}_{pg} = (^{n}\mathbf{M}_{g})^{-1}\, {}^{n}\mathbf{M}_{c}\, {}^{c}\mathbf{M}_{o}\, {}^{o}\mathbf{M}_{pg}$ where $^{n}\mathbf{M}_{g}$ and $^{n}\mathbf{M}_{c}$ are given by the robot's proprioceptive sensors.

Considering that the visual primitive is parameterized by $\mathbf{s} = (\mathbf{t}, \mathbf{u}\theta)$ where $\mathbf{t}$ is the position error between the current and desired frame, while $\mathbf{u}\theta$ is the orientation error, decomposed as the axis $\mathbf{u}$ and angle θ of the rotation between these two frames. The control law (3) is then applied using the Jacobian at the robot's gripper and the corresponding interaction matrix $\mathbf{L_s}$ given by:

$$\mathbf{L_s} = \begin{bmatrix} -\mathbf{I}_3 & [\mathbf{t}]_\times \\ \mathbf{0}_3 & \mathbf{L}_\omega \end{bmatrix} \tag{5}$$

where $\mathbf{I}_3$ and $\mathbf{0}_3$ are the 3×3 identity and zero matrices respectively, the $\mathbf{L}_\omega$ matrix is given by $\mathbf{L}_\omega = \mathbf{I}_3 - \frac{\theta}{2}[\mathbf{u}]_\times + \left(1 - \frac{sinc(\theta)}{sinc^2(\theta/2)}\right)[\mathbf{u}]_\times{}^2$, and $[\mathbf{t}]_\times$ is the skew symmetric matrix associated with vector $\mathbf{t}$.

4.5 Grasping Task

Along this task the robot's arm moves to grasp the desired item. The same number of DOF is constrained and the same technique is used to define the grasping task as in the previous case of pre-grasping, but the desired gripper pose is changed to the object's pose. Thus the task's error will be extracted from the relative pose between the gripper and the item ${}^{g}\mathbf{M}_{o}$ which is also regulated to zero (this pose is calculated using the relation: ${}^{g}\mathbf{M}_{o} = ({}^{n}\mathbf{M}_{g})^{-1}\,{}^{n}\mathbf{M}_{c}\,{}^{c}\mathbf{M}_{o}$.

After arriving to the desired pose, the robot's gripper closes to catch the item. We should note that the tasks are automatically sequenced and a task is completed once the error norm reaches a predefined threshold value. This threshold varies wrt. the executed task: during the grasping task of small objects, a high precision is necessary unlike in the case of pre-grasping task. In our experiments, the threshold is predefined to 0.8 mm in the camera's image for the visibility task. In case of pre-grasping and grasping tasks, the precision is predefined to 3 mm and 0.2 mm respectively and 2 deg for the orientation.

5 Experimental results

The presented tasks in the previous section have been implemented and tested several times on the Humanoid Nao robot to ensure the efficiency of this method. We used the 2 cameras embedded on the robot's head: the top one for localization and the bottom one for manipulation tasks. The control rate of the robot is equal to that of the camera (20 Hz) [2].

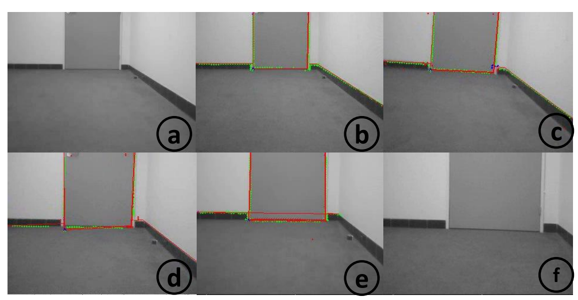
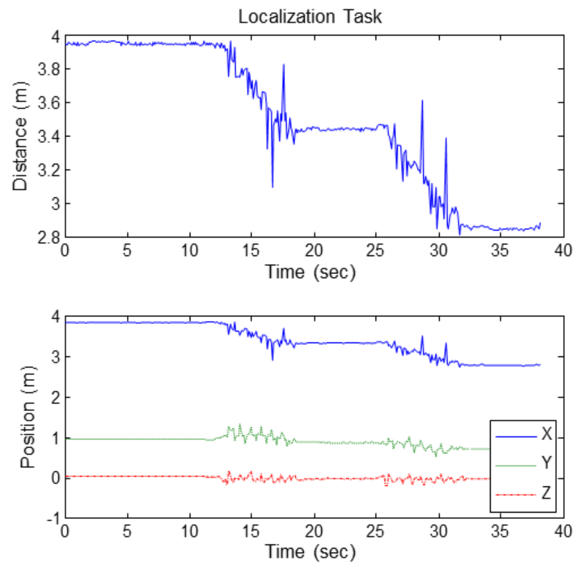


Fig. 4 Experiment photos of Localization Task during robot's locomotion

[2] See a video of the applied tasks on `www.youtu.be/WflA_jRBMkM`

Fig. 5 Experimental results of the localization task in $\mathscr{F}_n$ frame



5.1 Self-Localization Task Results

This part corresponds to the task defined in section 4.1. Fig. 4-a shows the robot's environment before launching the tracking and locomotion tasks. Fig. 4-b...e show the tracking of the door during the task's execution, and when the task is completed (Fig. 4-f). In Fig. 5, the distance between the robot's frame $\mathscr{F}_n$ and the origin of the door's frame is plotted, in addition to the X, Y and Z components of this pose in $\mathscr{F}_n$. The distance decreases from 3.94 m to 2.88 m. These results show then that the robot successfully tracks the door while walking the desired distance (1 m) with an error of 6 cm, which allow us to localize the robot.

5.2 Visibility and Grasping Task Results

Experiment photos of the grasping tasks executed by Nao robot are presented in Fig. 6 and correspond to the tasks presented in sections 4.2...4.5 above. Fig. 6-a, shows the item to grasp before launching the MBT to detect and track it (Fig. 6-b). The visibility task is used to center the object on the camera's image, and the pre-grasping task is executed in Fig. 6-c, where we can identify the different frames on the robot's arm and gripper in addition to the object's frame. Afterwards, the gripper's frame approaches the object's when executing the grasping task (Fig. 6-d). Finally the gripper closes and the manipulation task is completed (Fig. 6-e-f).

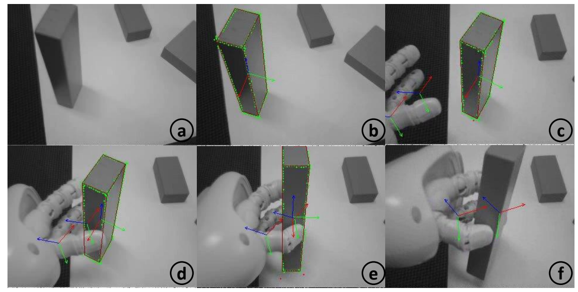


Fig. 6 Experiment photos of tracking and grasping of an item showing the arm's, gripper's and object's frames

The variation of the error in each task is presented in Fig. 7: the first graph represents the horizontal and vertical position error during the head servoing task (visibility task), initially the object is at a distance of approximately 30 cm from the center of the camera's image, we remark that this error is successfully regulated to zero during 18 sec with a precision of 0.8 mm. For the 2^{nd} and 3^{rd} graphs, we present the pre-grasping and grasping tasks errors on X, Y and Z components (in Nao's frame), and the Yaw angle of the gripper orientation: the robot's hand is initially at an approximate distance of 15 cm from the predefined pre-grasping position, and the gripper is rotated of 80 deg wrt. the object. During this task, the position error and the Yaw angle are regulated exponentially to zero during 30 sec. Finally, we recall that these (pre) grasping tasks are successfully executed with a precision of 3 mm and 0.2 mm respectively and 2 deg for the orientation.

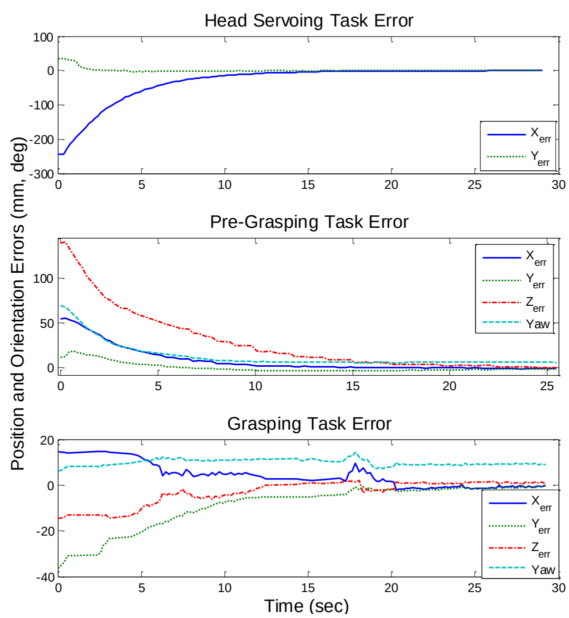


Fig. 7 Experimental results: Visibility task, Pre-Grasping task and Grasping task errors

6 Discussions and Conclusions

In this paper, we presented a concrete scenario of a humanoid mobile robot executing self-localization task while walking and manipulation tasks in an everyday life environment, and we detailed the elementary tasks used by the robot to perform this scenario. The experimental results point out the possibility and efficiency of using the MBT techniques to apply real-time 3D visual servoing on a simple humanoid robot (Nao) in case of localization and manipulation tasks.

For the self-localization task, the error may be relatively large wrt. other localization methods, but it is acceptable and sufficient when dealing with indoor locomotion for manipulation tasks, because of the high robustness of the MBT technique and the implemented automatic re-initialization of the tracking process. In Fig. 5, between the 10^{th} and 32^{nd} sec, we can identify some disturbances which are caused by the displacement of the camera during the robot's locomotion. During this period, the automatic re-initialization module of the tracking is used to prevent localization task from failure. To improve this method, an additional module could be used to dynamically compensate the camera's motions during robot's walking (using data from robot's accelerometer/gyrometer), or by fusion with other sensors data; for example using an external camera on the ceiling of the room to provide a wider view and more visual data, or the Nao's odometry module.

For the manipulation part, the results of the tracking and grasping tasks, which were tested several times, show that this method is robust to camera occlusion by the robot's hand, and robust to slight object movement due to hand-object collision. These tasks should be improved and tested on other robots. Indeed, the mechanic and software architecture of Nao is very constraining, especially for the small camera field of view, the inability of simultaneous use of the two cameras and the constrained operational space of the hands. For this, future works will concentrate on the improvement and implementation of this method on other platforms with different objects of complex shapes with mobile or articulated elements. Other sensors' feedbacks can also be used to improve the manipulation robustness and reactivity against dynamic or unusual changes in the environment.

References

1. Brock O, Kuffner J, Xiao J (2008) Motion for manipulation tasks. In: Springer Handbook of Robotics, Siciliano B and Khatib O (ed) Springer Berlin Heidelberg, pp. 615–645.
2. Evrard P, Mansard N, Stasse O, Kheddar A, Schau T, Weber C, Peer A, Buss M (2009) Intercontinental , multimodal , wide-range telecooperation using a humanoid robot. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS'09, pp. 5635–5640.
3. Nitzsche N, Schmidt G (2004) A mobile haptic interface mastering a mobile teleoperator. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS'04, vol. 4, sept-oct. 2004, pp. 3912–3917.

4. Houston K, Sieber A, Eder C, Vittorio O, Menciassi A, Dario P (2011) A teleoperation system with novel haptic device for micromanipulation. In: Int. Journal of Robotics and Automation, vol. 26, no. 3, 2011.
5. Miller W T (1987) Sensor-based control of robotic manipulators using a general learning algorithm. In: IEEE Journal of Robotics and Automation, vol. 3, no. 2, pp. 157–165, apr. 1987.
6. Dang H, Allen P (2010) Robot learning of everyday object manipulations via human demonstration. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS'10, oct. 2010, pp. 1284–1289.
7. Jain A, Kemp C (2009) Pulling open novel doors and drawers with equilibrium point control. In: IEEE-RAS Int. Conf. on Humanoid Robots, dec. 2009, pp. 498–505.
8. Mansard N, Stasse O, Chaumette F, Yokoi K (2007) Visually-guided grasping while walking on a humanoid robot. In: IEEE Int. Conf. on Robotics and Automation, apr. 2007, pp. 3041–3047.
9. Quinlan M J, Middleton R H (2009) Multiple model kalman filters: A localization technique for robocup soccer. In: RoboCup 2009: Robot Soccer World Cup XII, Baltes J, Lagoudakis M G, Naruse T, Ghidary S S (ed) Springer, 2010, vol. 5949, pp. 276–287.
10. Billington D, Estivill-Castro V, Hexel R, Rock A (2006) Using temporal consistency to improve robot localisation. In: RoboCup 2006: Robot Soccer World Cup X, Lakemeyer G, Sklar E, Sorrenti D, Takahashi T (ed) Springer Berlin/Heidelberg, 2007, vol. 4434, pp. 232–244.
11. Haverinen J, Kemppainen A (2009) Global indoor self-localization based on the ambient magnetic field. In: Int. Journal of Robotics and Autonomous Systems, vol. 57, no. 10, pp. 1028–1035, oct. 2009.
12. Thuilot B, Martinet P., Cordesses L, Gallice J (2002) Position based visual servoing: keeping the object in the field of vision. In: IEEE Int. Conf. on Robotics and Automation, vol. 2, 2002.
13. Gouaillier D, Hugel V, Blazevic P, Kilner C, Monceaux J, Lafourcade P, Marnier B et al (2009) Mechatronic design of nao humanoid. In: IEEE Int. Conf. on Robotics and Automation, may 2009, pp. 769–774.
14. Chaumette F, Hutchinson S (2006) Visual servo control. i. basic approaches. In: IEEE Robotics Automation Magazine, vol. 13, no. 4, pp. 82–90, dec. 2006.
15. Cervera E (2006) A Cross-Platform Network-Ready Visual Servo Simulator. In : IEEE Int. Conf. on Intelligent Robots and Systems (IROS'06), pp. 2314-2319, Beijing (China), 2006.
16. Marchand E, Spindler F, Chaumette F (2005) Visp for visual servoing: a generic software platform with a wide class of robot control skills. In: IEEE Robotics Automation Magazine, vol. 12, pp. 40–52, dec. 2005.
17. Comport A, Marchand E, Pressigout M, Chaumette F (2006) Real-time markerless tracking for augmented reality: the virtual visual servoing framework. In: IEEE Transactions on Visualization and Computer Graphics, vol. 12, no. 4, pp. 615–628, jul-aug. 2006.
18. Galindo C, Fernandez-Madrigal J-A, Gonzalez J, Saffiotti A (2008) Robot task planning using semantic maps. In: Robotics and Autonomous Systems, vol. 56, no. 11, pp. 955–966, 2008.
19. Stasse O, Verrelst B, Vanderborght B, Yokoi K (2009) Strategies for humanoid robots to dynamically walk over large obstacles. In: IEEE Transactions on Robotics, vol. 25, no. 4, pp. 960–967, aug. 2009.
20. Sorribes J, Prats M, Morales A (2007) Visual tracking of a jaw gripper based on articulated 3d models for grasping. In: IEEE Int. Conf. on Robotics and Automation, may 2010, pp. 2302–2307.