15

Safe Cooperation between Human Operators and Visually Controlled Industrial Manipulators

J. A. Corrales, G. J. Garcia, F. A. Candelas, J. Pomares and F. Torres
University of Alicante
Spain

1. Introduction

The development of industrial tasks where human operators and robots collaborate in order to perform a common goal is a challenging research topic on present robotics. Nowadays, industrial robots are isolated in fenced workspaces where human cannot enter in order to avoid collisions. However, this configuration misses the opportunity of developing more flexible industrial tasks where humans and robots work together. This collaboration takes advantage of the complementary features of both entities. On the one hand, industrial robots are able to perform repetitive tasks which can be exhausting and monotonous for human operators. On the other hand, humans are able to perform specialized tasks which require intelligence or dexterity. Thereby, industrial tasks can be improved substantially by making humans and robots collaborate in the same workspace. The main goal of the work described in this chapter is the development of a human-robot interaction system which enables this collaboration and guarantees the safety of the human. This system is composed of two subsystems: the human tracking system and the robot control system.

The human tracking system deals with the precise real-time localization of the human operator in the industrial environment. It is composed of two systems: an inertial motion capture system and an Ultra-WideBand localization system. On the one hand, the inertial motion capture system is composed of 18 IMUs (Inertial Measurement Units) which are attached to the body of the human operator. These sensors obtain precise orientation measurements of the limbs of the body of the operator and thus full-body movements of the operator are tracked. On the other hand, the Ultra-WideBand localization system obtains a precise measurement of the global position of the human operator in the environment. The measurements of both systems are combined with a Kalman filter algorithm.

Thereby, all the limbs of the body of the human operator are positioned precisely in the environment while the human-robot interaction task is performed. These measurements are applied over a skeleton which represents the basic structure of the human body. However, this representation by itself does not take into account the actual dimensions of the surface of the body because each limb is modelled as a line. The bones of this skeleton have been covered with bounding volumes whose dimensions match approximately the size of the corresponding human limbs. The implemented bounding volumes have been organized in a three-level hierarchy in order to reduce the computational cost of the distance computation.

The robot control system is based on visual servoing. Most current industrial robots are controlled only with kinematic information without permitting the interaction of the robot

Source: Human-Robot Interaction, Book edited by: Daisuke Chugo,
ISBN 978-953-307-051-3, pp. 288, February 2010, INTECH, Croatia, downloaded from SCIYO.COM

with its environment. Nevertheless, the robot needs sensors in order to adapt its behaviour depending on the changes in the environment when human-robot interaction tasks are performed. In this work, the robot has a camera mounted at its end-effector to control its movements according to a visual servoing method. The main objective of visual servoing is to minimize the error between the image obtained at the first pose of the robot and the image obtained at the target position of the robot. Visual servoing is adequate to control the robot in situations where external or not planned objects enter the robot workspace, avoiding a possible collision. When the robot has to perform a planned path in a 3D space limited by other objects, classic image based visual servoing fails to track this planned path. Previous research on visual path tracking has tried to solve this problem by making use of visual servoing to follow a desired image path previously sampled in time. These systems can be modified in order to obtain a human safety algorithm. In this way, when a human is dangerously close to the robot, the path tracking must be stopped. However, after the danger of collision has disappeared, previous image trajectory tracking methods based on visual servoing fail to return to the initial path because they are time-dependent. Therefore, a time-independent behaviour of the control system is crucial to develop interactions with the workspace. This time-independent behaviour makes the robot continue on the same path from the same point that it was tracking before the detection of the human. The method described in this chapter guarantees the correct tracking in the 3D space at a constant desired velocity.

As shown before, a safety behaviour which stops the normal path tracking of the robot is performed when the robot and the human are too close. This safety behaviour has been implemented through a multi-threaded software architecture in order to share information between both systems. Thereby, the localization measurements obtained by the human tracking system are processed by the robot control system to compute the minimum human-robot distance and determine if the safety behaviour must be activated.

This chapter is organized as follows. Section 2 describes the human tracking system which is used to localize precisely all the limbs of the human operator who collaborates with the robotic manipulator. Section 3 presents an introduction to visual servoing and shows how the robot is controlled by a time-independent path tracker based on it. Section 4 describes how the safety behaviour which avoids any collision between the human and the robot is implemented. This section presents the hierarchy of bounding volumes which is used to compute the human-robot distance and modify the movements of the robot accordingly. Section 5 enumerates the results obtained in the application of all these techniques in a real human-robot interaction task where a fridge is disassembled. Finally, the last section presents the conclusions of the developed research.

2. Human tracking system

2.1 Components of the human tracking system

The human operator who interacts with robotic manipulators has to be localized precisely in the industrial workplace because of two main reasons. On the one hand, the knowledge of the human location enables the development of safety behaviours which avoid any risk for the physical integrity of the human while their interaction takes place. On the other hand, the localization of the human operator can also be taken into account to modify the movements of the robot accordingly. The movements of the robot are adapted to the human's behaviour and thus more flexible human-robot interaction tasks can be implemented.

The necessary precision of the human localization system depends on the type of interaction task and the distance between the human and the robot. For instance, in interaction tasks where the human operator and the robot collaborate on the assembly of big sized products, a global localization system which only obtains a general position of the human and the robot is sufficient because they work far from each other. Nevertheless, in most interaction tasks, the robot and the human need to collaborate in close distances and a localization of their limbs and links is required. The position of each link of the robotic manipulator can be easily obtained from the robot controller through forward kinematics. However, an additional motion capture system is necessary to register the movements of all the limbs of the human operator.

In this chapter, an inertial motion capture suit (*GypsyGyro-18* from *Animazoo*) has been used to localize precisely all the limbs of the human operator. This system has several advantages over other motion capture technologies (Welch & Foxlin, 2002): it does not suffer from magnetic interferences (unlike magnetic systems), optical occlusions do not affect its precision (unlike vision systems) and it is comfortable to wear (unlike mechanical systems). The main component of this inertial system is a lycra suit with 18 IMUs (Inertial Measurement Units) which is worn by the human operator. Each IMU is composed of 3 MEMS (Micro-Electro-Mechanical Systems) gyroscopes, 3 accelerometers and 3 magnetometers. The measurements from these 9 sensors are combined by a complementary Kalman filter (Foxlin, 1996) in order to obtain the orientation (relative rotation angles: roll, pitch and yaw) of the limb to which the IMU is attached. This inertial motion capture system not only registers the relative orientations of all the limbs of the human's body but it also computes an estimation of the global displacement of the human through a foot-step extrapolation algorithm. Nevertheless, this algorithm sometimes does not detect correctly when a new step takes place and this fact involves the accumulation of a small global position error (drift) which becomes excessive after several steps (Corrales et al., 2008).

An additional global localization system based on UWB signals (*Ubisense v.1* from *Ubisense*) has been used to solve this problem and correct the error accumulated by the inertial motion capture system in the global positioning of the human. This localization system is based on two different devices: four sensors which are installed at fixed positions of the industrial workplace and a small tag which is worn by the human operator. This small tag sends UWB pulses which are processed by the four sensors in order to compute an estimation of the global position of the tag in the environment by implementing a combination of AoA (Angle of Arrival) and TDoA (Time-Difference of Arrival) techniques. Thereby, this UWB system obtains precise estimates of the global position of the human operator. The global position measurements of both systems are combined through the fusion algorithm described in the following section.

2.2 Fusion algorithm

The two systems (inertial motion capture system and UWB system) which compose the developed human tracking system have complementary features which show the suitability of their combination. On the one hand, the inertial motion capture system registers precise relative limbs orientations with a high sampling rate (30 - 120Hz). However, the global position estimated by this system is prone to accumulate drift. On the other hand, the UWB localization system calculates a more precise global position of the human operator but with a considerably lower sampling rate (5 - 10Hz). Furthermore, the measurements of the UWB

system can be easily related to fixed objects in the environment because they are represented in a static coordinate system while the measurements of the inertial motion capture system are represented in a dynamic coordinate system which is established every time the system is initialized. All these complementary features have been taken into account in order to develop a fusion algorithm which estimates precisely the position of the human operator from the position measurements of both tracking systems (Corrales et al., 2008). The reader can see Table 1 for a detailed description of the implemented fusion algorithm. The limbs orientation measurements from the inertial motion capture system are not processed by this algorithm because they are very precise and do not need any correction process.

```
01: Initialize ᵁT_G with the first two measurements: p_1^G , p_2^U
02: for each measurement p_t
03:   if p_t is from the inertial motion capture system G
04:     if p_t-1 is from the UWB localization system U
05:       Recalculate ᵁT_G with p_t and x_t-1
06:     end if
07:     Transform p_t from G to U with ᵁT_G
08:     x_t =KalmanFilterPrediction( p_t )
09:   else if p_t is from the UWB localization system U
10:     x_t =KalmanFilterCorrection( p_t , x_t-1 )
11:   end if
12: end for
```

Table 1. Pseudocode of the fusion algorithm based on a Kalman filter.

First of all, the global position measurements registered by both tracking systems have to be represented in the same coordinate system. The static coordinate system *U* of the UWB system has been chosen as reference system and thus the inertial motion capture measurements have to be transformed from their frame *G* to it. The first two measurements of both systems are used to compute the transformation matrix ${}^{U}\mathbf{T}_{G}$ between their coordinate systems (line 1 of Table 1). If the inertial motion capture system did not have any errors, this initial value of the transformation matrix would always be valid. Nevertheless, as the inertial motion capture measurements accumulate a drift; this transformation matrix has to be updated accordingly. This update process is based on a Kalman filter (Thrun et al., 2005) which aims to reduce the accumulated drift.

The state of the implemented Kalman filter is composed by the 3D position $\mathbf{x}_t = (x_t, y_t, z_t)$ of the human operator. Each time a measurement from one of the tracking systems is received, one of the steps of the Kalman filter (prediction or correction) is executed. The global position measurements $\mathbf{p}_t^G$ of the inertial motion capture system are applied in the prediction step of the Kalman filter (line 8 of Table 1) while the global position measurements $\mathbf{p}_t^U$ of the UWB localization system are applied in the correction step (line 10 of Table 1). An estimate $\hat{\mathbf{x}}_t$ of the position of the human operator is obtained from each of these filter steps. In addition, the transformation matrix ${}^{U}\mathbf{T}_{G}$ is recalculated after each correction step of the filter (line 5 of Table 1). Thereby, the drift accumulated by the inertial motion capture system is corrected because the next measurements of this system are transformed with this new transformation matrix (line 7 of Table 1).

The structure of the developed fusion algorithm takes the most of the complementary features of both tracking systems. On the one hand, the application of the measurements of the motion capture system in the prediction step of the filter maintains their high sampling rate and enables the tracking of quick movements of the human. On the other hand, the application of the measurements of the UWB system in the correction step of the filter removes the drift accumulated by the previous measurements of the motion capture system. Thereby, the resulting tracking system from this fusion algorithm has the high sampling rate of the motion capture system and the precision of the UWB system.

3. Robot control system

Nowadays, the great majority of repetitive assembly or disassembly tasks are commonly developed in the industry by robot manipulators. These tasks are usually defined by a set of poses for the robot manipulator. The robot is isolated and its workspace is constant. The robot is frequently controlled kinematically by different poses that it has to perform in order to complete the task. Nevertheless, when the robot workspace is not constant, the robot needs additional information in order to react to any unexpected situation. This is the case we are dealing with in this chapter. Sight allows the human brain to perceive the shapes, the colours and the movements. Computer vision permits the robot to obtain important information about a changing environment. Visual servoing is a technique that controls the robot movements using the visual information processed by a computer vision system. In Section 3.1 an introduction of visual servoing is presented. Unfortunately, classic visual servoing systems are only adequate for placing the robot in a relative position between it and an object in the environment. In this case, the 3D path that the robot follows to arrive to this goal position is not controlled in any way. When this 3D path followed by the robot must be controlled, classic visual servoing controllers have to be modified in order to precisely track the predefined path. These modifications are described in section 3.2.

3.1 Introduction to visual servoing

Visual servoing is a technique which uses visual information to control the motion of a robot (Hutchinson et al., 1996). It is a technique widely developed in the literature in the last two decades. There are two basic types of visual servoing: the image-based visual servoing and the position-based visual servoing. Image-based visual servoing uses only visual features extracted from the acquired images, $\mathbf{s}$, to control the robot. Therefore, these controllers do not need neither a complete 3D model of the scene nor a perfect camera calibration. The desired visual features, $\mathbf{s}_d$, are obtained from a desired final position of the robot in the scene. Image-based visual servoing controllers minimize the error between any robot position and the goal position by minimizing the error of the visual features computed from the images acquired at each robot position, $\mathbf{e} = (\mathbf{s} - \mathbf{s}_d)$. To minimize exponentially this error $\mathbf{e}$, a proportional control law is used:

$$\dot{\mathbf{e}} = -\lambda \mathbf{e} \tag{1}$$

where λ is a proportional gain.

In a basic image-based visual servoing approach, the velocity of the camera, $\mathbf{v}_c$ is the command input for controlling the robot movements. To obtain the control law, the interaction matrix, $\mathbf{L}_s$, must be firstly presented. The interaction matrix is a matrix that

relates the variations of the visual features in the image with the variations of the poses of the camera in the 3D space, i.e. its velocity (Chaumette & Hutchinson, 2006).

$$\dot{\mathbf{s}} = \mathbf{L}_s \mathbf{v}_c \tag{2}$$

From Equation (1) and Equation (2), the velocity of the camera to minimize exponentially the error in the image is obtained:

$$\mathbf{v}_c = -\lambda \hat{\mathbf{L}}_s^+ (\mathbf{s} - \mathbf{s}_d) \tag{3}$$

where $\hat{\mathbf{L}}_s^+$ is the pseudoinverse of an approximation of the interaction matrix. This camera velocity is then transformed to obtain the velocity to be applied to the end-effector of the robot. To do this, the constant relation between the camera and the end-effector is used in a camera-in-hand configuration (i.e., the camera is mounted at the end-effector of the robot). If the system uses a camera-to-hand configuration, the relation between the robot base and the camera is usually known, and the forward kinematics provides the relation between the robot base and the end-effector coordinate systems.

Image-based visual servoing systems present singularities and/or local minima problems in large displacements tasks (Chaumette & Hutchinson, 2006). To overcome these drawbacks maintaining the good properties of image-based visual servoing robustness with regard to modeling and camera calibration errors, the tracking of a sufficiently sampled path between the two distant poses can be performed.

3.2 Time-independent visual servoing path tracking

Path planning is a commonly studied topic in visual servoing (Fioravanti, 2008). However, the technique used to perform the tracking of the planned image path is not usually presented. The research effort is usually focused on planning the trajectory of the visual features in the image. The main objective of this planning is to avoid the outliers features or the robot joint limits. Only some of the systems proposed up to now to plan trajectories in the image using visual servoing present the technique chosen to perform the tracking (Chesi & Hung, 2007; Malis, 2004; Mezouar & Chaumette, 2002; Pomares & Torres, 2005; Schramm & Morel, 2006). Most of them resolve the problem by sampling the path in the time (i.e., the trajectory is generated from a path and a time law) (Chesi & Hung, 2007; Malis, 2004; Mezouar & Chaumette, 2002). These systems employ a temporal reference, $\mathbf{s}_d(t)$:

$$\mathbf{v}_c = -\lambda \hat{\mathbf{L}}_s^+ (\mathbf{s} - \mathbf{s}_d(t)) \tag{4}$$

This control action is similar to the one employed by an image-based visual servoing (3). However, in (4) image features of the desired trajectory are obtained from a time-dependent function $\mathbf{s}_d(t)$. These systems do not guarantee the correct tracking of the path when the robot interacts with its environment. If the robot interacts with an object placed in its workspace, the system continues sending visual references to it even though the robot cannot move. Once the obstruction ends, the references that have been sent up to that moment are lost, not allowing the correct tracking of the path. A robot controller with this time-dependent behaviour is not valid for a global human-robot interaction system. When the safety system takes part in the task because the human and the robot are too near, the robot has to move away from the human and the time-dependent visual servoing system loses the references until the distance is safe again.

Only a few tracking systems based on visual servoing have been found in the literature with a time-independent behaviour (Schramm & Morel, 2006; Pomares & Torres, 2005). However, it is not possible to specify the desired velocity at which the robot tracks the trajectory with these previous visual servoing systems. In particular, (Schramm & Morel, 2006) present a visual tracker which reduces the tracking velocity to zero at every intermediate reference of the tracked trajectory. The movement flow-based visual servoing by (Pomares & Torres, 2005) does not stop the robot at each intermediate reference but it does not guarantee a minimum desired tracking velocity. Furthermore, this method suffers from great oscillations when the tracking velocity is increased. The proposed time-independent visual servoing system overcomes this limitation and the tracking velocity can be adjusted to a desired value $|\mathbf{v}_d|$. In this system, the desired visual features do not depend on the time. They are computed depending on the current position of the camera, $\mathbf{s}_d(\mathbf{q})$. This type of visual servoing does not lose any reference and thus, the robot is able to follow the complete path once the human-robot distance is safe again.

In the research exposed in this chapter, the planning path issue is not relevant. There are a lot of works related with this topic. The robot control system must be provided with a desired image path, and this is obtained here in a previous off-line stage. Before the tracking process, the discrete trajectory of the features in the image to be tracked by the robot is sampled $\boldsymbol{T} = \{ {}^k\mathbf{s} / k \in 1 \ldots N \}$, with ${}^k\mathbf{s}$ being the set of M points or features observed by the camera at instant k, ${}^k\mathbf{s} = \{ {}^k f_i / i \in 1 \ldots M \}$. For instance, Fig. 1 shows the desired image trajectory that the robot has to track to accomplish the task. This path stores the set of M image features (four laser points) which are observed by the camera at each instant k, ${}^k\mathbf{s}$.

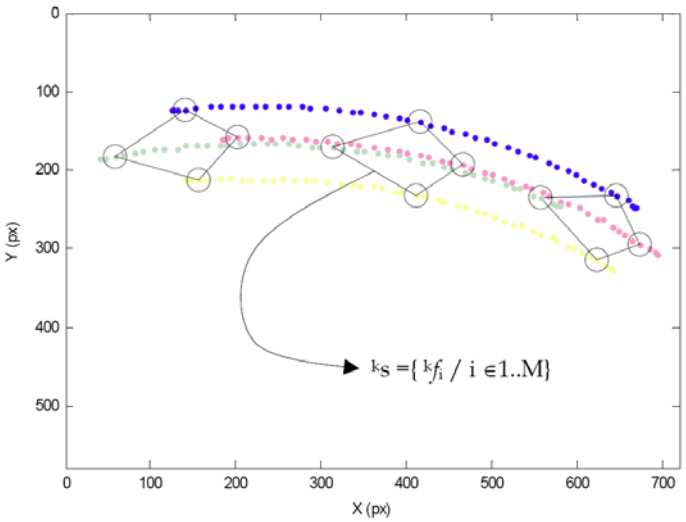


Fig. 1. Image trajectory to be tracked during the disassembly task.

The set of visual features observed at the initial camera position are represented by ${}^1\mathbf{s}$. From this initial set of image features, it is necessary to find an image configuration which provides the robot with the desired velocity $|\mathbf{v}_d|$ by iterating over the set $\boldsymbol{T}$. For each image configuration ${}^k\mathbf{s}$, the corresponding camera velocity is determined considering an image-based visual servoing system (at the first step $\mathbf{s} = {}^1\mathbf{s}$):

$$^{k}\mathbf{v} = -\lambda \hat{\mathbf{L}}_{s}^{+}\left(\mathbf{s} - {}^{k}\mathbf{s}\right) \tag{6}$$

This process continues until the velocity $|^{k}\mathbf{v}|$ which is the nearest to $|\mathbf{v}_{d}|$ is obtained.
At this moment, the set of features $^{k}\mathbf{s} = {}^{i}\mathbf{s}$ will be the desired features to be used by an image-based visual servoing system (see Equation (3)) in order to at least maintain the desired velocity. However, the visual features which provide the exact desired velocity are between $^{k}\mathbf{s}$ and $^{k-1}\mathbf{s}$. To obtain the correct image features the interpolation method described in (Garcia et al., 2009b) is proposed. This interpolation method searches for a valid camera configuration between the poses of the camera in the k and k-1 image path references. To reconstruct the 3D pose of the camera from which the $^{k}\mathbf{s}$ and $^{k-1}\mathbf{s}$ sets of visual features are observed, virtual visual servoing is employed, taking advantage of all the images acquired until that moment and all the background of this technique to calibrate the camera during the visual servoing task (Marchand & Chaumette, 2001).
Therefore, once the control law represented in Equation (3) is executed, the system searches again for a new image configuration which provides the desired velocity. This process continues until the complete trajectory is tracked.

4. Human-robot integration

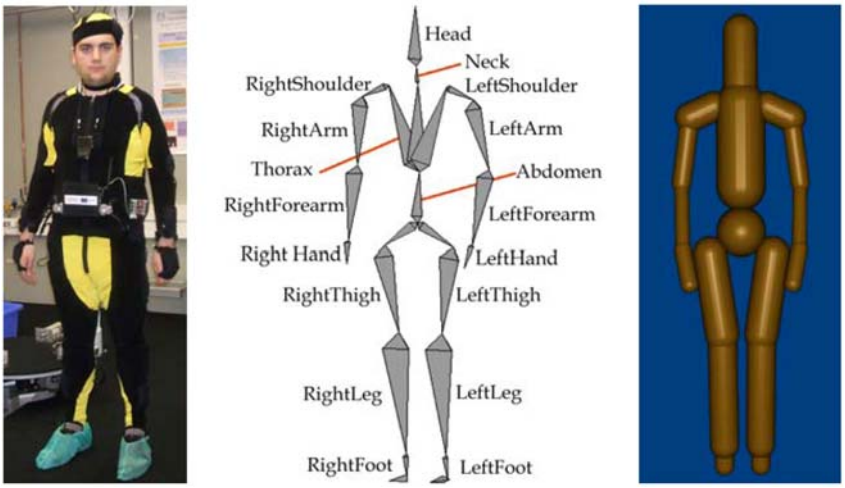
4.1 Human-robot interaction behaviour

The human-robot interaction behaviour implemented in this chapter is based on two main components: a hierarchy of bounding volumes and a safety strategy. The hierarchy of bounding volumes is used to model approximately the bodies of the human operator and the robot. The safety strategy computes the minimum distance between these bounding volumes and changes the behaviour of the robot accordingly. In the following sub-sections both elements are described in detail.

4.1.1 Hierarchy of bounding volumes

As it has been stated before, all the limbs of the human operator and all the links of the robotic manipulator have to be localized precisely in order to assure the safety of the human and develop flexible human-robot interaction tasks. The tracking system described in section 2 combines the measurements of an inertial motion capture suit and a UWB localization system (see Fig. 2a) in order to obtain the orientation of all the limbs of the human operator and her/his global position in the environment. All these measurements are applied over a skeleton structure (see Fig. 2b) which represents the kinematic structure (links and joints) of the human's body. The positions of the two ends of each link of this skeleton can be calculated by applying forward kinematics to the measurements of the tracking system.
However, this skeletal representation only considers the length of each link but it does not take into account the 3D dimensions of the link's surface. Therefore, an additional geometric representation is necessary to model the surface of the human's body and localize the human operator completely and precisely. This geometric representation has to be based on bounding volumes because a detailed mesh representation based on polygons would be too expensive for real-time operation. The selected bounding volume should fulfill two requirements: tight fitting to the links' surface and inexpensive distance computation. Previous similar human-robot interaction systems (Balan & Bone, 2006; Martinez-Salvador et al., 2003) implement sphere-based geometric representations to achieve this goal. These

sphere-based representations have an inexpensive distance computation but they do not fit tightly to the links' surface. A high number of spheres are needed to solve this problem and thus the final computational efficiency of this geometric representation is reduced.



(a) (b) (c)

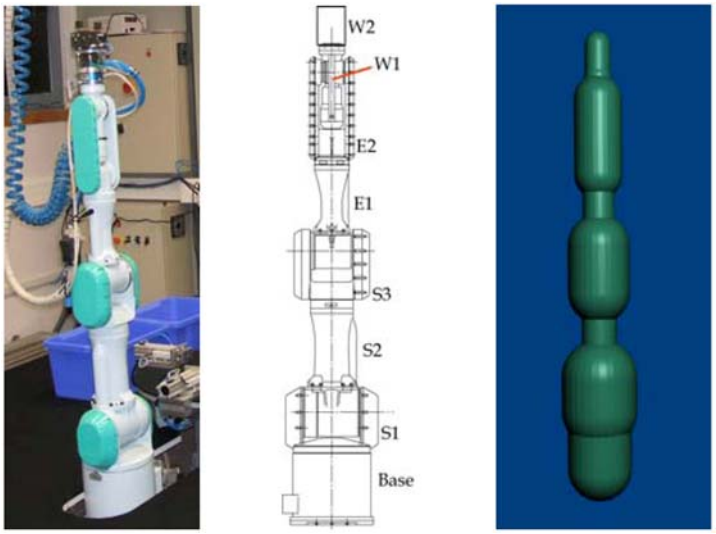
Fig. 2. Model for the human operator: (a) Human operator with the tracking system; (b) Links of the human; (c) SSL bounding volumes which cover the links of the human.

In this chapter, a new bounding volume representation based on Sphere-Swept Lines (SSLs) is presented to overcome the limitations of previous spherical models. A SSL is a bounding volume obtained from the Minkowski sum of a sphere and a segment (Ericson, 2005). SSLs have a better tight than spheres because they have a similar shape to the limbs. In addition, the distance computation between two SSLs has a low cost because it is calculated from the difference between the SSLs' radii and the distance between the inner segments of the SSLs (Schneider & Eberly, 2003). The geometric representation of the human operator is computed by covering each of the links of the skeleton obtained from the tracking system with a SSL (see Fig. 2c). The location of each SSL is updated on real-time by matching the two ends of the inner segment of the SSL with the ends of the corresponding link.

The robotic manipulator (see Fig. 3a) which collaborates with the human operator is modelled in a similar way. The positions of the links of the robot (see Fig. 3b) are computed by applying forward kinematics to the joint values registered by the robot controller. However, as in the case of the human operator, this procedure only obtains the locations of the ends of each link but it does not consider its surface. A SSL will cover each link in order to model all the dimensions of it.

This geometric representation consists of 18 SSLs for the human operator and 8 SSLs for the robotic manipulator. This fact implies that each time the minimum distance between the human and the robot is computed, 144 pairwise distance tests are required. In order to reduce this number of distance tests and increase the performance of the distance computation, a three-level hierarchy of bounding volumes has been developed. Fig. 4 and Fig. 6 depict the relations between the components of this hierarchy for the human operator

and the robotic manipulator, respectively. Fig. 5 and Fig. 7 show the 3D representation of the bounding volumes which compose each level of this hierarchy for the human operator and the robotic manipulator, respectively.



(a) (b) (c)

Fig. 3. Model for the robotic manipulator: (a) PA-10 manipulator; (b) Links of the robot; (c) SSL bounding volumes which cover the links of the robot.

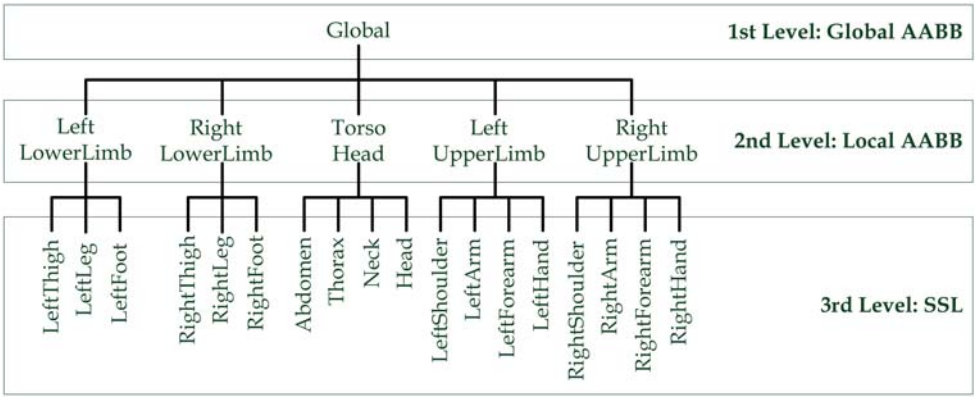
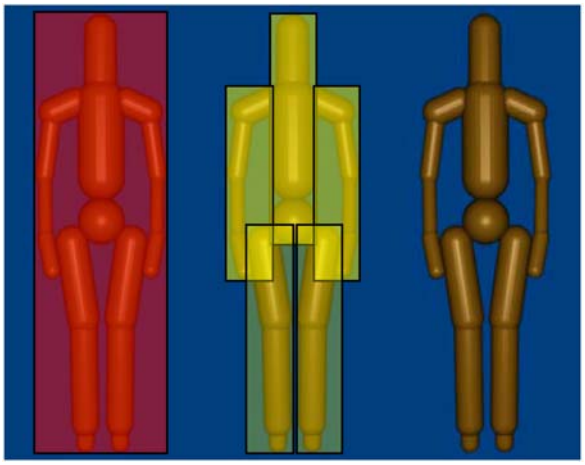


Fig. 4. Hierarchy of bounding volumes which cover the body of the human operator.

The third level of this hierarchy is composed by the SSL bounding volumes described above (see Fig. 2c and Fig. 3c) and it will only be used when the human operator and the manipulator are working quite close to each other. A distance threshold will be established in order to determine whether the third level or the second level of this hierarchy have to be applied to obtain the minimum human-robot distance. The second level of the bounding volume hierarchy is composed by AABBs (Axis-Aligned Bounding Boxes) which cover



(a) (b) (c)

Fig. 5. 3D representation of the hierarchy of bounding volumes for the human operator: (a) Global AABB (level 1); (b) Local AABBs (level 2) and (c) SSLs (level 3).

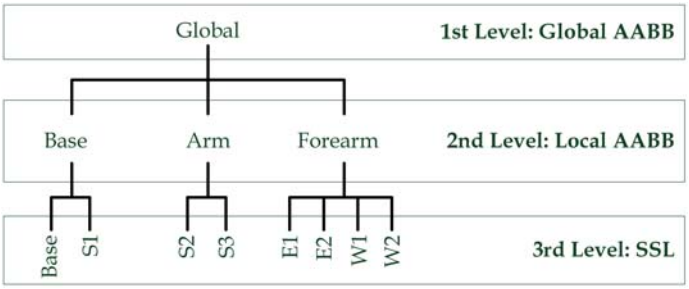
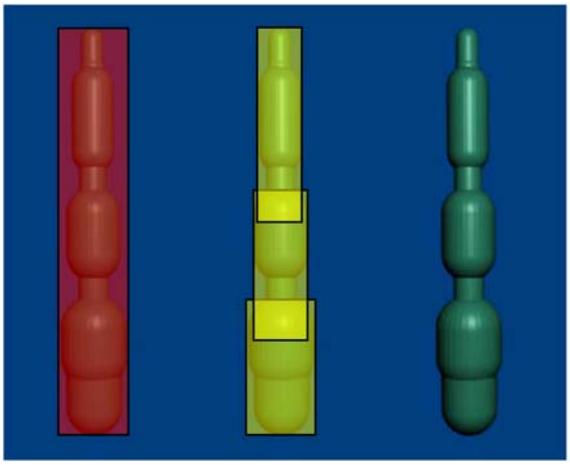


Fig. 6. Diagram of the hierarchy of bounding volumes for the robotic manipulator.

several SSLs of the third level that are related kinematically. In particular, the second level of the bounding volume hierarchy for the human operator is composed by 5 AABBs (left lower limb, right lower limb, torso-head, left upper limb and right upper limb) and by 3 AABBs (base, arm and forearm) for the robot. Therefore, the number of pairwise distance tests is reduced to 15 for the second level of the hierarchy. The AABBs of this level are computed by looking for the maximum and minimum coordinates of the contained links and adding to them the maximum radius of the contained SSLs. The first level of this hierarchy is composed by one global AABB which covers the bodies of the human or the robot and which is computed from the maximum and minimum coordinates of all the links. This global AABB will be used when the human and the robot are far from each other. In a similar way to the second and third levels, a distance threshold will be established in order to determine if the first or the second level of the bounding hierarchy is used to compute the human-robot distance.

The selection of the hierarchy level to be used depends on the distance between the human and the robot. When the distance between the human operator and the robot is high, a geometric representation based on AABBs (levels 1 or 2 of the hierarchy) is used. This



(a) (b) (c)

Fig. 7. 3D representation of the hierarchy of bounding volumes for the robotic manipulator: (a) Global AABB (level 1); (b) Local AABBs (level 2) and (c) SSLs (level 3).

representation increases the efficiency of the distance computation because the number of distance pairwise tests is reduced and AABB-AABB distance tests are less expensive than SSL-SSL distance tests. When the human-robot distance is small, the SSL bounding volumes are necessary to compute a good approximation of the distance because they are a more precise and detailed representation of the human and robot's links. However, as stated before, the SSL-based representation requires more pairwise distance tests. In order to reduce the number of SSL pairwise distance tests, a combination of the levels 2 and 3 of the hierarchy is implemented for the minimum distance computation. This minimum distance algorithm and the processing of the obtained value for modifying the robot behaviour are explained in detail in the following section.

4.1.2 Safety strategy for human-robot interaction

The main goal of the hierarchy of bounding volumes described in the previous section is to represent the bodies of the human operator and the robot in such a way that the human-robot distance computation is precise and efficient for the development of cooperative tasks on real-time. The algorithm which has been implemented to compute the human-robot distance is shown in Table 2.

The first step of this algorithm (line 1 of Table 2) is to initialize the two distance thresholds: $\mathbf{DIST_{1>2}}$ and $\mathbf{DIST_{2>3}}$. The first threshold $\mathbf{DIST_{1>2}}$ determines whether the minimum human-robot distance is obtained from the AABBs of the first level or from the AABBs of the second level. Similarly, the second threshold $\mathbf{DIST_{2>3}}$ determines whether the minimum human-robot distance is obtained from the AABBs of the second level or from the SSLs of the third level. After having initialized the thresholds, this algorithm generates the two global AABBs of the first level (lines 2 and 3 of Table 2) by looking for the minimum and maximum coordinate values of all the links. Next, the minimum distance **mdist1** between both global AABBs is calculated (line 4 of Table 2) and compared with the first threshold $\mathbf{DIST_{1>2}}$. If **mdist1** is bigger than the threshold, this distance value will be used as minimum human-

robot distance (line 6 of Table 2) because they are so far from each other that the global AABB representation is sufficient.

However, if the human-robot distance is smaller than the threshold $\mathbf{DIST_{1>2}}$, the more precise AABB representation of the second level is required. First of all, the local AABBs of the second level are generated (lines 8 and 9 of Table 2) from the minimum and maximum coordinate values of the links contained in each AABB. As there are 5 AABBs for the human and 3 AABBs for the robot, they are stored in two arrays of these lengths: $\mathbf{AABB2_H[1..5]}$ and $\mathbf{AABB2_R[1..3]}$. All the possible distance values between each pair of AABBs are computed and stored in array **dist2[1..15]** (line 10 of Table 2). This array is sorted in ascending order (line 11 of Table 2) and its minimum component is used as the minimum distance value **mdist2** for the second level of the bounding volume hierarchy (line 12 of Table 2). This value **mdist2** is used as minimum human-robot distance if it is higher than the second threshold $\mathbf{DIST_{2>3}}$ (line 14 of Table 2).

```
01: Initialize distance thresholds: DIST1>2, DIST2>3
02: Compute Global AABB (Level 1) for Human: AABB1H
03: Compute Global AABB (Level 1) for Robot: AABB1R
04: mdist1= MinimumDistance (AABB1H, AABB1R)
05: if (mdist1 > DIST1>2)
06:   finalDist= mdist1 //Distance in Level 1
07: else
08:   Compute Local AABBs (Level 2) for Human: AABB2H[1..5]
09:   Compute Local AABBs (Level 2) for Robot: AABB2R[1..3]
10:   dist2[1..15]= PairwiseDistances(AABB2H[1..5], AABB2R[1..3])
11:   dist2[1..15]= SortArrayInAscendingOrder(dist2[1..15])
12:   mdist2= MinimumValue(dist2[1..15])
13:   if (mdist2 > DIST2>3)
14:     finalDist= mdist2 // Distance in Level 2
15:   else
16:     mdist3= FLOAT_MAX_VALUE
17:     for each element i in dist2[1..15]
18:       if (mdist3 < dist2[i])
19:         break //Finish for loop in line 17
20:       end if
21:       Compute SSLs (Level 3) contained by AABB2H[i]: SSL3H[1..nLinksHi]
22:       Compute SSLs (Level 3) contained by AABB2R[i]: SSL3R[1..nLinksRi]
23:       for each element j in SSL3H[1..nLinksHi]
24:         for each element k in SSL3R[1..nLinksRi]
25:           distSSL= MinimumDistance(SSL3H[j], SSL3R[k])
26:           mdist3= MinimumValue(distSSL, mdist3)
27:         end for
28:       end for
29:     end for
30:     finalDist= mdist3 //Distance in Level 3
31:   end if
32: end if
33: return finalDist
```

Table 2. Pseudocode of the minimum distance algorithm based on the three-level hierarchy of bounding volumes.

When **mdist2** is smaller than $\mathbf{DIST_{2>3}}$, the SSL representation is applied because AABB bounding volumes do not provide a sufficient level of detail for precise distance computation between very close links. In fact, the SSL bounding volumes have a better tight to the links than the AABB bounding volumes. In order to reduce the number of distance tests between each pair of SSLs, the distance values **dist2[1..15]** between the AABBs of the second level are used as a lower threshold for the SSLs distance values. The SSLs are generated (lines 21 and 22 of Table 2) from the corresponding AABBs of the second level, which have been ordered according to their pairwise distances (line 11 of Table 2). After generating the SSLs contained by each pair of AABBs, the minimum distance **mdist3** between them is computed (lines 25 and 26 of Table 2). If the SSL-SSL distance **mdist3** becomes smaller than the distance **dist2[i]** between the following pair of AABBs (line 18 of Table 2), the minimum distance search process will end (line 19 of Table 2) because the following SSLs will be contained by those AABBs and they will be further away. Thereby, in most cases, not all the 144 SSL-SSL tests are required.
The minimum distance value computed by the algorithm presented in Table 2 is used by the safety strategy in order to guarantee that there are no collisions between the human operator and the robot. In particular, the safety strategy verifies that the human-robot distance is higher than a safety threshold. While this condition is fulfilled, the robot tracks its trajectory normally. Nevertheless, when the human-robot distance is lower than the safety threshold, the robot tracking process is temporarily stopped and a safety behaviour is activated. This safety behaviour moves the robot away from the human operator in order to maintain the human-robot distance above the safety threshold.

4.2 Software architecture

A multi-threaded software architecture has been developed in order to implement the human-robot interaction behaviour described in the previous sections. It has been programmed as a C++ program which is executed in the controller PC. The robot controller, the motion capture system, the UWB localization system and the vision system are connected to this PC in order to avoid any synchronisation problem between their measurements.
Three threads compose this software architecture (see Fig. 8): the distance computation thread, the path tracking thread and the safety strategy thread. These three threads are executed simultaneously and share a common memory space where they interchange information. The distance computation thread obtains the links orientation measurements from the human tracking system (see Section 2) and the joints angles of the robot from the robot controller. The positions of the links of the human and the robot are computed from these measurements through a forward kinematics algorithm and they are stored in the common memory space. Finally, these links positions are used by the algorithm described in Table 2 in order to generate the hierarchy of bounding volumes and calculate the minimum distance between the human operator and the robot manipulator. The distance computation thread will update this human-robot distance value each time new measurements from the human tracking system and the robot controller are registered by the controller PC.
The minimum human-robot distance calculated by the distance computation thread is stored in the common memory space where it is checked by the other two threads. On the one hand, when this human-robot distance is greater than the safety threshold, the path tracking thread performs the visual servoing path tracking process (see Section 3.2). The

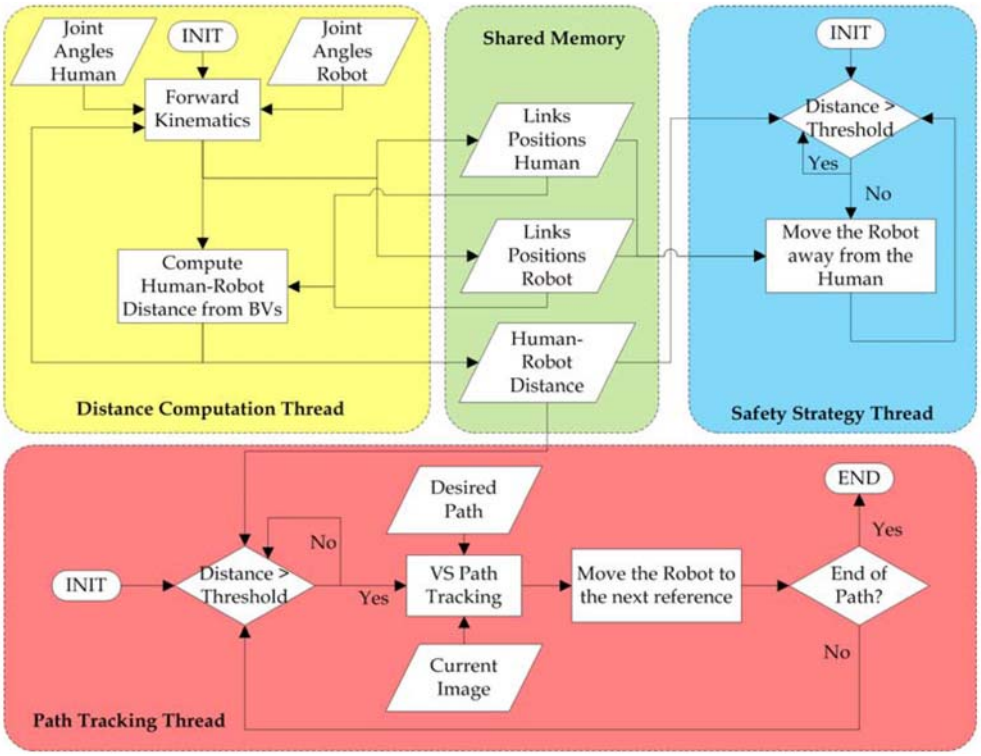


Fig. 8. Components of the software architecture which implements the human-robot interaction behaviour.

time-independent path tracker compares the current image from the eye-in-hand camera with the corresponding image of the desired path and calculates the robot velocity required to make them coincide. This tracking process ends when the desired path is completed and is paused when the safety behaviour is activated. On the other hand, when the human-robot distance is smaller than the safety threshold, the safety strategy thread executes the safety behaviour. This safety behaviour moves the robot away from the human operator in order to keep the human-robot distance above the safety threshold. This escape trajectory is calculated from the line which links the two closest links. While the safety behaviour is executed, the time-independent path tracking process is paused, and vice versa.

5. Experimental results

5.1 Task description

The human-robot interaction system proposed here can be applied in any collaborative task where the human enters in the robot workspace. In particular, the described system has been applied to a disassembly task. The object to be disassembled is a small fridge. The main elements which take part in this task are depicted in Fig. 9: the fridge, the Mitsubishi PA-10 robotic manipulator which unscrews the fridge lid, the human operator who extracts the internal tray and the storage box where the different parts of the fridge are stored after they

are disassembled. The Mitsubishi PA-10 robotic manipulator has three devices installed at its end-effector in order to perform the task: a screwdriver, a JR3 force sensor and an eye-in-hand PHOTONFOCUS MV-D752-160-CL-8 camera. The camera is able to acquire and to process up to 100fps using an image resolution of 320x240px. The image trajectory is generated by using four laser points projected on the floor as extrated features for the visual servoing path tracking system. The human operator wears an *Animazoo* inertial motion capture suit and an *Ubisense* Ultra-WideBand (UWB) radio localization system in order to track precisely all her/his movements and compute the human-robot distance with the bounding volume hierarchy.

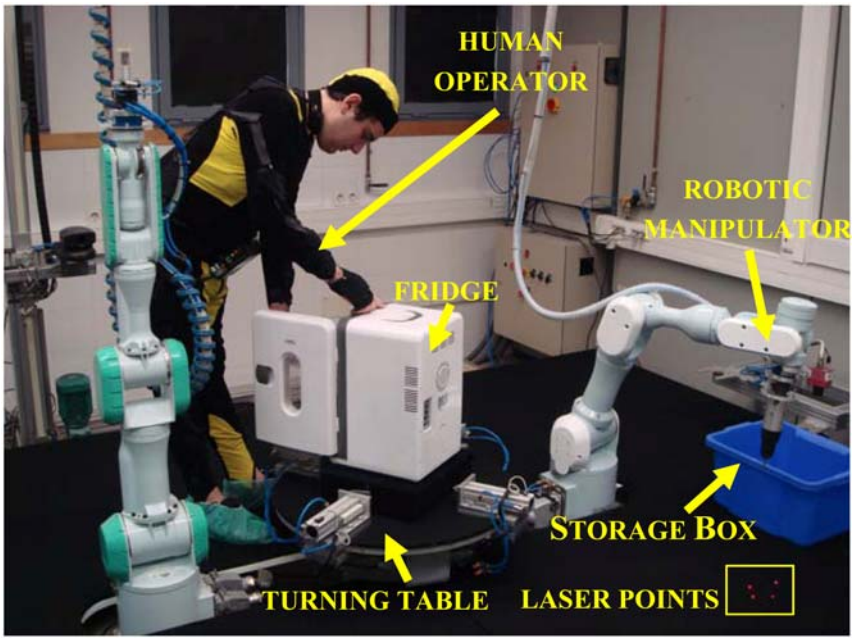


Fig. 9. Experimental setup for the fridge disassembly task.

The disassembly task can be split in the following subtasks to be performed by the human operator and the robot:

- Robot: The robotic manipulator has to remove the screws from the rear lid of the fridge. Firstly, the robotic manipulator goes from the storage box to the unscrew position by tracking a predefined path using the image path tracker based on visual servoing described in Section 3.2. Secondly, the robot unscrews the corresponding screw and then the robot is again guided to the storage box where the screw is left. This task is repeated until it removes all the screws.
- Human operator: Meanwhile, the human operator has to empty the fridge's contents.

The tasks described can be performed simultaneously. The global safety system presented in this chapter is employed during all the disassembly task. In Fig. 10, the sequence of the disassembly task is depicted. The image path tracker guides the robot from the storage box to the next screw (Frame 1 of Fig. 10). During the tracking, the human approaches the fridge in order to empty its content (Frame 2 of Fig. 10). The distance between the human operator

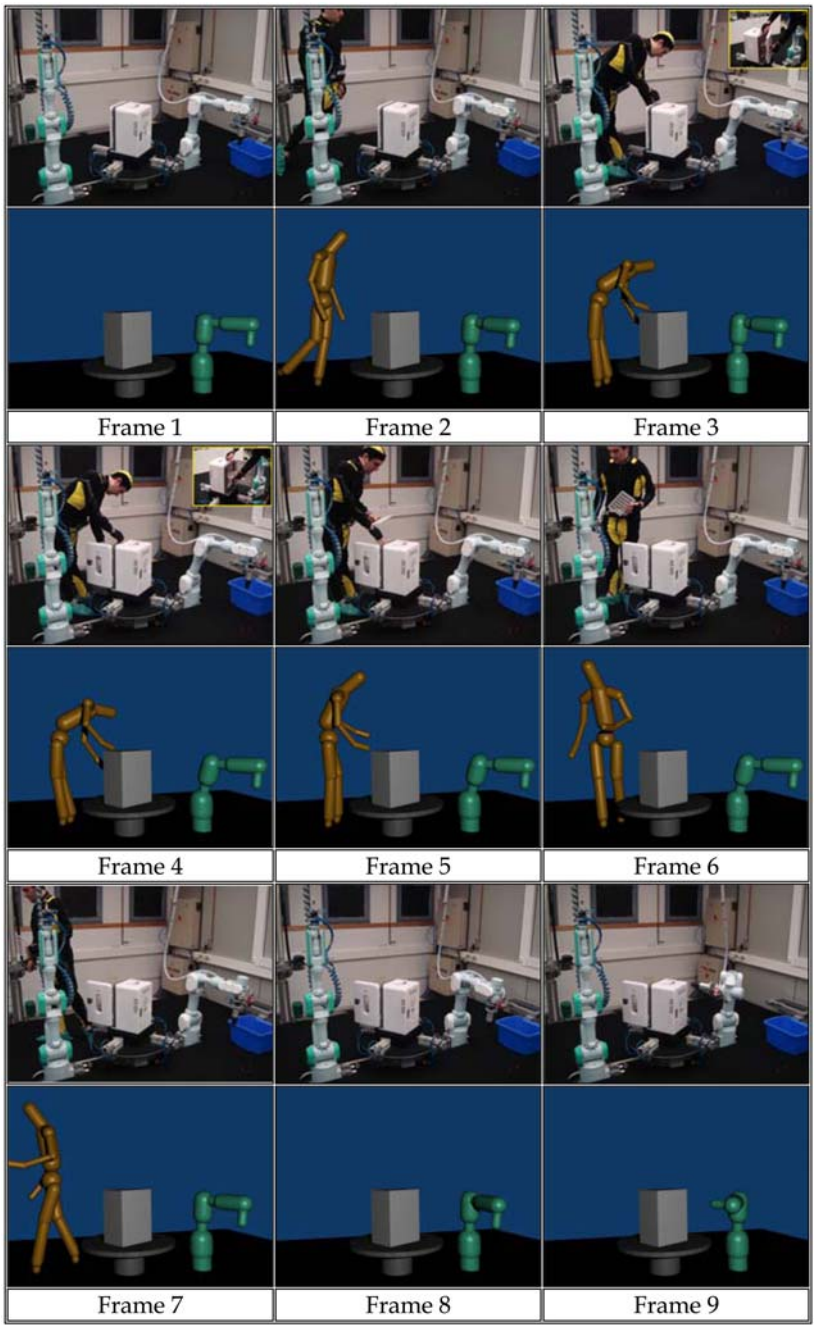


Fig. 10. Disassembly sequence with human-robot interaction. Each frame of the sequence is shown with a photograph of the workspace and the corresponding SSLs bounding volumes.

and the robot falls under the safety threshold activating the safety strategy. The robot goes away from the human operator (Frames 3 and 4 of Fig. 10) and stops until the distance exceeds the safety threshold again. Meanwhile, the human operator opens the fridge's door (Frame 4 of Fig. 10) and takes the internal tray out of the fridge (Frames 5 and 6 of Fig. 10) in order to carry it to a storage box which is out of the workspace. While the human operator is going away from the robot, the human-robot distance is again greater than the safety threshold and the visual servoing path tracking is re-activated (Frames 7 and 8 of Fig. 10). Thanks to the time-independent behaviour, the path is then tracked correctly and the robot can arrive at the unscrew position by following the predefined path (Frame 9 of Fig. 10).

5.2 Robot controller results

To show the correctness of the proposed time-independent image path tracker described in Section 3.2, a comparative between this time-independent and a time-dependent systems is next presented. Fig. 11 shows the evolution of the features in the image obtained with the two different methods. Both of the methods perform the tracking correctly until the moment when the obstruction begins. Nevertheless, from the moment when the robot is released, the time-dependent system is not able to return to the exact point in the trajectory where the obstruction began. This is due to the loss of temporal references. Therefore, the time-independent method described in this chapter is adequate for the tracking of image paths in tasks where the robot interacts with a human or with any object in its workspace.

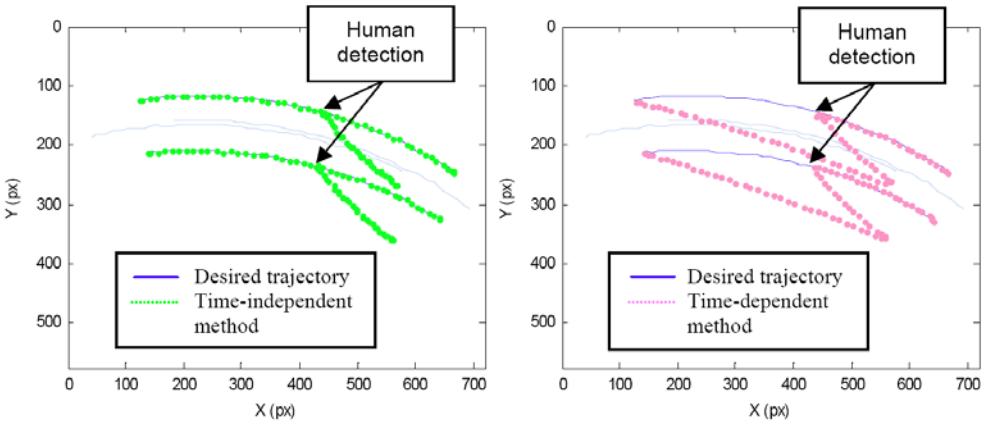


Fig. 11. (a) Evolution (from right to left)of the features in the image space using the time-independent tracking; (b) Evolution (from right to left) of the features in the image using a time-dependent method.

5.3 Human-robot integration results

The main goal of the safety strategy is to maintain the human-robot distance above a safety threshold (0.5m for this disassembly task). This safety strategy calculates the human-robot distance on real-time by executing the algorithm in Table 2 based on the three-level hierarchy of bounding volumes described in Section 4.1.1. The distance threshold $\mathbf{DIST_{1>2}}$ of this algorithm is set to 2m while the distance threshold $\mathbf{DIST_{2>3}}$ is set to 1m. Therefore, the

global AABBs (level 1) are used for distances greater than 2m; the local AABBs (level 2) are used for the distances between 2m and 1m and the SSLs (level 3) are used for distances smaller than 1m.

Fig. 12 depicts the evolution of the minimum human-robot distance obtained by the distance algorithm for the disassembly task. This plot shows how the human operator approaches the robot while the robot performs the time-independent visual servoing path tracking from iteration 1 to iteration 289. In iteration 290, the safety strategy starts and the robot controller pauses the path tracking. The safety strategy is executed from iteration 290 to iteration 449 and it tries to keep the human-robot distance above the safety threshold (0.5m). In iteration 450, the robot controller re-activates path tracking because the human-robot distance is again greater than the threshold when the human is going away from the workspace.

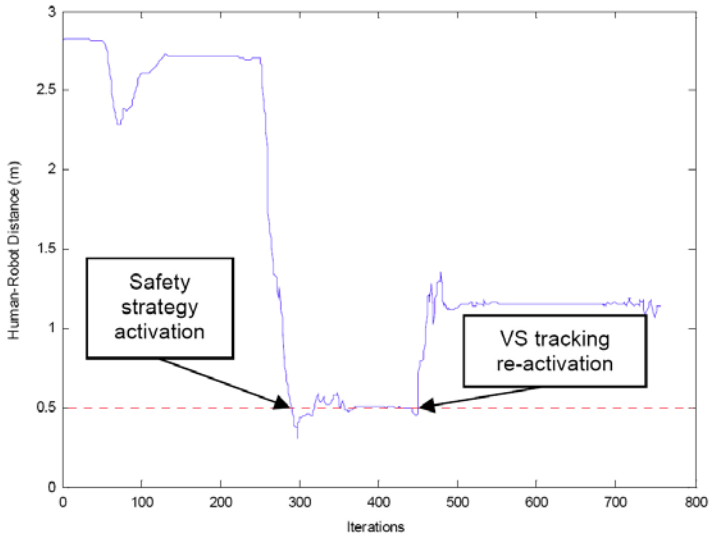


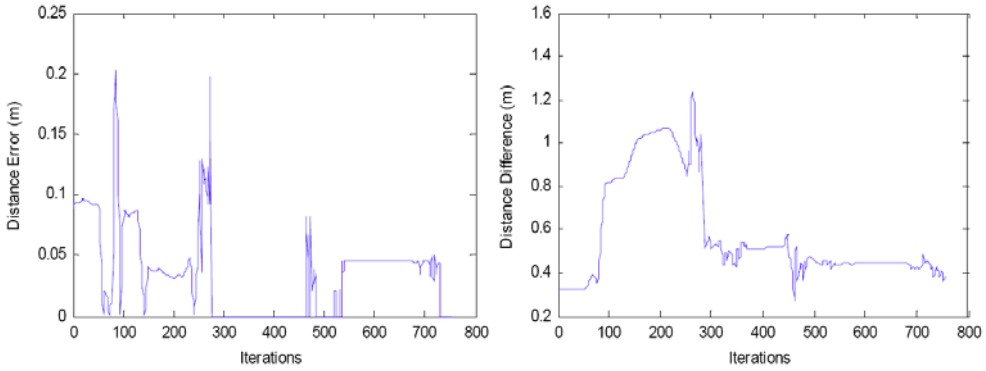
Fig. 12. Evolution of the minimum human-robot distance during the disassembly task.

Fig. 13.a. depicts the error evolution of the distance obtained by the algorithm in Table 2 with regard to the distance values obtained from the SSL bounding volumes, which are used as ground-truth. This figure shows an assumable mean error of 4.6cm for distances greater than 1m. For distances smaller than 1m (between iterations 280 and 459), the error is null because the SSLs are used for the distance computation.

The proposed distance algorithm obtains more precise distance values than previous research. In particular, Fig. 13.b. shows the difference between the distance values obtained by the algorithm in Table 2 and the distance values computed by the algorithm in (Garcia et al., 2009a), where no bounding volumes are generated and only the end-effector of the robot is taken into account for the distance computation instead of all its links.

Fig. 14 shows the histogram of distance tests which are performed for the distance computation during the disassembly task. In 64% of the executions of the distance algorithm, a reduced number of pairwise distance tests is required (between 1 and 16 tests) because the bounding volumes of the first and/or second level of the hierarchy (AABBs) are used. In the remaining 36%, between 30 and 90 distance tests are executed for the third level

of the hierarchy (SSLs). This fact demonstrates that the hierarchy of bounding volumes involves a significant reduction of the computational cost of the distance computation with regard to a pairwise strategy where 144 distance tests would always be executed.



(a) (b)

Fig. 13. (a) Evolution of the distance error from the BV hierarchy; (b) Evolution of the distance difference between the BV hierarchy algorithm and (Garcia et al., 2009a).

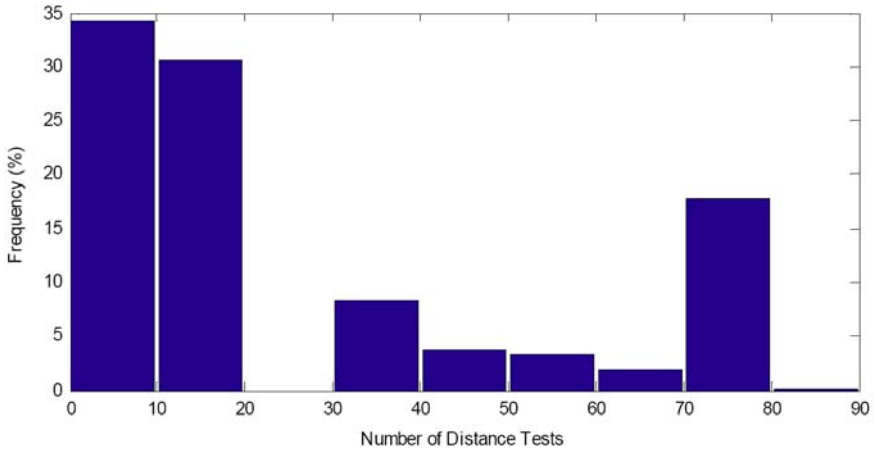


Fig. 14. Histogram of the number of distance tests required for the minimum human-robot distance computation.

6. Conclusions

This chapter presents a new human-robot interaction system which is composed by two main sub-systems: the robot control system and the human tracking system. The robot control system uses a time-independent visual servoing path tracker in order to guide the movements of the robot. This method guarantees that the robot tracks the desired path completely even when unexpected events happen. The human tracking system combines the measurements from two localization systems (an inertial motion capture suit and a UWB

localization system) by a Kalman filter. Thereby, this tracking system calculates a precise estimation of the position of all the limbs of the human operator who collaborates with the robot in the task.

In addition, both sub-systems have been related by a safety behaviour which guarantees that no collisions between the human and the robot will take place. This safety behaviour computes precisely the human-robot distance by a new distance algorithm based on a three-level hierarchy of bounding volumes. If the computed distance is below a safety threshold, the robot's path tracking process is paused and a safety strategy which tries to maintain this separation distance is executed. When the human-robot distance is again safe, the path tracking is re-activated at the same point where it was stopped because of its time-independent behaviour. The authors are currently working at improving different aspects of the system. In particular, they are considering the use of dynamic SSL bounding volumes and the development of more flexible tasks where the human's movements are interpreted.

7. Acknowledgements

The authors want to express their gratitude to the Spanish Ministry of Science and Innovation and the Spanish Ministry of Education for their financial support through the projects DPI2005-06222 and DPI2008-02647 and the grant AP2005-1458.

8. References

Balan, L. & Bone, G. M. (2006). Real-time 3D collision avoidance method for safe human and robot coexistence, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 276-282, Beijing, China, Oct. 2006.

Chaumette, F. & Hutchinson, S. (2006). Visual Servo Control, Part I: Basic Approaches. *IEEE Robotics and Automation Magazine*, Vol. 13, No. 4, 82-90, ISSN: 1070-9932.

Chesi, G. & Hung, Y. S. (2007). Global path-planning for constrained and optimal visual servoing. *IEEE Transactions on Robotics*, Vol. 23, 1050-1060, ISSN: 1552-3098.

Corrales, J. A., Candelas, F. A. & Torres, F. (2008). Hybrid tracking of human operators using IMU/UWB data fusion by a Kalman filter, *Proceedings of 3rd. ACM/IEEE International Conference on Human-Robot Interaction,* pp. 193-200, Amsterdam, March 2008.

Ericson, C. (2005). *Real-time collision detection*, Elsevier, ISBN: 1-55860-732-3, San Francisco, USA.

Fioravanti, D. (2008). Path planning for image based visual servoing. Thesis.

Foxlin, E. (1996). Inertial head-tracker sensor fusion by a complementary separate-bias Kalman filter, *Proceedings of IEEE Virtual Reality Annual International Symposium*, pp. 185-194, Santa Clara, California, 1996.

Garcia, G.J., Corrales, J.A., Pomares, J., Candelas, F.A. & Torres, F. (2009). Visual servoing path tracking for safe human-robot interaction, *Proceedings of IEEE International Conference on Mechatronics*, pp. 1-6, Malaga, Spain, April 2009.

Garcia, G.J., Pomares, J. & Torres, F. (2009). Automatic robotic tasks in unstructured environments using an image path tracker. *Control Engineering Practice*, Vol. 17, No. 5, May 2009, 597-608, ISSN: 0967-0661.

Hutchinson, S., Hager, G. D. & Corke, P. I. (1996). A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 5, 651-670, ISSN: 1042-296X.

Malis, E. (2004). Visual servoing invariant to changes in camera-intrinsic parameters. *IEEE Transactions on Robotics and Automation*, Vol. 20, No. 1, February 2004, 72-81, ISSN: 1042-296X.

Marchand, E. & Chaumette, F. (2001). A new formulation for non-linear camera calibration using VVS. Publication Interne 1366, IRISA, Rennes, France.

Martinez-Salvador, B., Perez-Francisco, M. & Del Pobil, A. P. (2003). Collision detection between robot arms and people. *Journal of Intelligent and Robotic Systems*, Vol. 38, No. 1, Sept. 2003, 105-119, ISSN: 0921-0296.

Mezouar, Y. & Chaumette, F. (2002). Path planning for robust image-based control. *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 4, 534-549, ISSN : 1042-296X.

Pomares, J. & Torres, F. (2005). Movement-flow based visual servoing and force control fusion for manipulation tasks in unstructured environments. *IEEE Transactions on Systems, Man, and Cybernetics – Part C*, Vol. 35, No. 1, 4-15, ISSN: 1094-6977.

Schneider, P. J. & Eberly, D. H. (2003). *Geometric tools for computer graphics,* Elsevier, ISBN: 1-55860-594-0, San Francisco, USA.

Schramm, F. & Morel, G. (2006). Ensuring visibility in calibration-free path planning for image-based visual servoing. *IEEE Transactions on Robotics*, Vol. 22 No. 4, 848-854, ISSN : 1552-3098.

Thrun, S., Burgard, W. & Fox, D. (2005). *Probabilistic Robotics*, MIT Press, ISBN: 978-0-262-20162-9, Cambridge, USA.

Welch, G., & Foxlin, E. (2002). Motion tracking: no silver bullet but a respectable arsenal. *IEEE Computer Graphics and Applications*, Vol. 22, No. 6, Nov. 2002, 24-38, ISSN: 0272-1716.