An Evaluation of Camera Pose Methods for an Augmented Reality System: Application to Teaching Industrial Robots

Abstract: In automotive industry, industrial robots are widely used in production lines for many tasks such as welding, painting or assembly. Their use requires, from users, both a good manipulation and robot control. Recently, new tools have been developed to realize fast and accurate trajectories in many production sectors by using the real prototype of vehicle or a generalized design within a virtual simulation platform. However, many issues could be considered in these cases: the delay between the design of the vehicle and its production is often important, moreover, the virtual modeling presents a non realistic aspect of the real robot and vehicle, so this factor could introduce localization inacurracies in performing trajectories. Our work is registered as a part of TRI project (Teleteaching Industrial Robots) which aims to realize a demonstrator showing the interaction of industrial robots with virtual components and allowing to train users to realize successfully their tasks on a virtual representation of a production entity.

In this project we make use of Augmented Reality (AR) techniques to overlay virtual objects onto the real world in order to enhance the user's perception and interaction while performing a specific industrial task. The pose accuracy is prerequisite of our application since it allows a reliable teaching of the real trajectory. Therefore, we survey some vision-based pose computation algorithms and present a method that offers increased robustness and accuracy in the context of real-time AR tracking. Our aim is to determine the performance of these pose estimation methods in term of errors and distance evaluation. The evaluation of the pose estimation methods was obtained using a series of tests and an experimental protocol. The analysis of results shows the performance of algorithms in term of accuracy, stability and convergence.

Keywords: Augmented Reality; pose estimation; industrial robot; computer vision; real time tracking.

1 Introduction

In recent years, robotics industry has seen considerable advances in technologies development, significant progress has been made to realize high performance systems. Industrial robotic systems are increasingly being used in factories, and a growing need of robot services is becoming undeniable in these environments. These trends droved pressing requirements of identifying new ways of programming robots safely, quickly and more intuitively. These methods should focus on robot service and address Human-Robot Interaction issues in industrial robotics.

Several methods exist for programming robots involving teaching them a sequence of points to define a trajectory. The programming process is dependent on time, errors and often requires several trials before the required accuracy is getting acceptable.

By introducing AR technologies in this programming process, the operator gets instant real-time visual feedback of a simulated process in relation to the real object performed in reduced programming time and increased quality of the resulting robot program. The strengths of AR over conventional methods include flexibility in providing visual guidance to the user during the programming process in various environments without the need to model the environment entities, and offers at the same time increased intuition and efficiency in the robot programming process.

In literature many works focused on the use of AR technologies in industrial context. The major application of AR systems in industry includes teaching, assistance and maintenance in order to provide an effective tool for operators to perform complex tasks in short time, increase their level of intuitiveness in the process and obtain the necessary flexibility needed today.

In [20], Pettersen et al. presented a demonstrator of a standalone AR pilot system allowing an operator to program robot waypoints and process specific events related to paint applications. The system presents visual feedback of the paint result for the operator, allowing him to inspect the process result before the robot has performed the actual task. The main purpose of this system is to develop a robot programming system that is easy to use, speeds up the programming process, utilizes the intuitive process knowledge of the operator, and increases the quality of the finished program without the need of a CAD model. This programming method using AR to visualize the paint result has proved to be faster than traditional robot programming methods. Additionally, the operators report that they find the process of programming robots much easier and intuitive.

Chong et al. [6] explored the potential of teaching a robot to perform an arc welding task in an AR environment. The authors introduced a method to define as robot teaching in AR, a virtual robot model is rendered onto a tracked marker and the teaching is performed using a physical probe with an attached marker. The system demonstrated the potential of AR for teaching robots in general. The main feature of the system is that the user is able to plan a task for the robot in an unstructured and unmodeled environment, in a relatively short time with assistance from visual feedback provided in the AR environment. However, the developed system requires comfortable and easy interaction methods and the accuracy of the performed task have to be considered and studied.

In [3], Biegelbauer et al. presented the FlexPaint project which aims to automate robot programming applications. A new approach is reported to automatically generate the painting motion of industrial robots. The approach uses a sensing and painting cell where the part geometry is acquired, relevant features are extracted and the corresponding paint routines are grouped to obtain optimal painting trajectories. First implementations at industrial users show that the approach is feasible. However, the developed system needs improvements, automatic painting assumes a nonmoving part and the avoidance of robot singularities while performing the painting task.

Ong et al. [19] presented the potential of using an AR environment to facilitate immersive robot programming in unknown environments. The benefits of an AR environment over conventional robot programming approaches are discussed, followed by a description of the robot programming using AR system and new methodologies for programming robotic tasks. The immersive robot programming system allows the user to move directly a virtual robot in an unknown environment. The major system issue that needs to be investigated, is the level of accuracy achievable using AR system. The authors expected that the use of more sophisticated and accurate tracking systems would significantly improve user acceptance of these methodologies.

In [22], a registration evaluation Mixed Reality (MR) system using an industrial robot is described. In this evaluation system, the tip of the robot arm plays the role of the user's head, where a head mounted display is mounted. By using an industrial robot, the camera pose with a high level of accuracy and robustness is obtained. Additionally, the system gives the the ability to play back the same specified operations repeatedly under identical conditions. The authors implemented the system and proposed evaluation methods for motion robustness, distance robustness, jitter, etc. and verified the validity of their system through some experiments. The system could be improved by evaluating the application in larger space and also record natural human motions and have a robot replay the motions. On the other hand there are some evaluation criteria were not introduced in this work like robustness to occlusion, lighting environment, video noise, and other complex ambient factors which should also be considered as evaluation criteria.

Bischoff et al. [4] showed the interest of AR in industrial application, the advantage and the potential of AR techniques are presented and demonstrated how they improve human robot interaction. The authors realized a first prototype of KUKA AR Viewer which includes various visualization and simulation options and allows instantaneous and real-time visual feedback. The developed system confirmed that AR is especially useful for robot training by adding synthetic graphics to enhance visualization of coordinate systems, robot motions and path information within the real robot cell. In addition, such a system allows robot motions simulation before its actual execution and gain an understanding for using the different virtual graphics to enhance users perception.

Shimizu et al. [23] presented a robotic user interface combined with MR technology to enable the presentation of enhanced visual information of a robot existing in the real world. The authors proposed the virtual kinematics to enhance robot motion, A MR system with virtual kinematics presented a selection of visual information by controlling the robot through physical simulation and by changing the parameter dynamically.

From the related work described before, we notice that the reliability of the developed technologies depends mostly on the accuracy of the used AR system. Indeed, these systems based AR techniques enhance visual information by superimposing virtual graphics on image sequences. The robustness and the accuracy of such a visual system is closely related to the determination of the transformation describing the relationship between coordinate frames (pose) of the camera and the object template used as target in the tracking process.

The camera pose estimation is an important step for tracking in AR applications. It allows the projection of synthetic models at the right location on real images. AR environments in which synthetic objects are inserted into a real scene, is a prime candidate since a potentially restricted workspace demands robust and fast pose estimation from few feature points. Several approaches are formulated to solve the camera pose parameters. The problem is considered as a nonlinear minimization and it is solved by least squares methods or nonlinear optimization algorithms, typically, the Gauss-Newton [2] [14] or Levenberg-Marquardt method [18]. Most solutions are iterative and depend on nonlinear optimization of some geometric constraints, either on the world coordinates or on the projections to the image plane. For real-time applications, linear or closed-form free from initialization are the most used solutions [17].

Dhome et al. [8] developed an analytical pose estimation method based on the interpretation of a triplet of any image lines and on the search of the model attitude. Ansar and Daniilidis [1] estimated the camera pose from an image of n points or lines with known correspondences. The authors presented a general framework which allows the pose estimation for both n points and n lines. Lu et al. [15] developed a fast and globally convergent pose estimation algorithm, called, Orthogonal Iteration (OI). The pose estimation problem is formulated as problem of error minimization based on object collinearity in image space. In [16], Maidi et al. used an Extended Kalman Filter (EKF) to estimate the transformation between the object and the camera coordinate frames. Based on the knowledge of the feature point position in the camera frame, the perspective projection matrix of the camera is computed and solved using the two steps of the EKF. Maidi et al. [17] developed a new pose estimation algorithm based on a combination of an analytical and an iterative method. An EKF is used to perform a nonlinear optimization of pose parameters which were initialized by an analytical algorithm.

Several methods based photogrammetry and using closed-form solutions for 3 points were developed in the literature [9] [10] [11] [12]. Quan and Lan [21] proposed a family of linear methods that yield a unique solution to 4 and 5 point pose determination for generic reference points. The authors showed that their methods do not degenerate for coplanar configurations and even outperform the special linear algorithm for coplanar configurations in practice.

In this work, our primary interest is to contribute to the improvement of pose accuracy for an AR system used as a waypoints teaching tool for industrial robots. We will make a performance comparative study between different pose estimators. The retained method must be accurate, stable, and respect real-time constraints using 4 coplanar matching 2D/3D points. In this work, we assume that the motion of the camera or the target object are unpredictable.

The remainder of the paper is organized as follows. In section 2, we present the context of work and the main problematic of our study. Then in section 3, we give an overview of the proposed solution. Section 4 describes the different modules constituting our system. We detail in section 5 the implemented solution. Section 6 presents the experimental setup and shows the obtained results. A discussion is presented in section 7 and we finish by section 8 where we present conclusion and future work.

2 Context and problematic of work

In the automotive industry a considerable waist of time exists between the moment of vehicle design and when it is setting into serial lines production in factories. This production is generally performed by automated robotic systems and a delay is required for the development and the automation of tools intended to program robot tasks for new vehicle model.

Currently there are only two ways to make a trajectory teaching on a robot: using a CAD or using manual teaching by an operator. In CAD techniques the programing is performed entirely in a virtual environment and this generates significant shifts comparing to reality. The virtual robot is a perfect design and presents no shortcomings, consequently, large gaps are created when trajectory points are transfered to the real robot. These differences are due to the fact that the virtual robot is not a faithful representation of the real robot (backlash and mechanical wear which could not be simulated in the virtual world). The other major drawback of this method is that the fittings movements on board the robot (cables, pipes, covers, etc.) can not be simulated in CAD, that may cause interference and collisions with the real part for trajectory transfer on the real robot (despite possible alterations). In addition, the robot cycle time calculated by CAD is approximate because it is related to the computer CPU sampling frequency which differs from the robot CPU frequency.

Concerning the manual learning, the manual programming has the disadvantage of being an approximate programming performed visually which requires continual alterations during the workpiece life. Moreover, this technique requires the presence of the actual workpiece to be able to perform teaching. Finally, operations must be made near the robot which may causes collision risk between the robot and the operator.

The problematic of our work is how to mix so closely geometric modeling, kinematics and dynamics of a real robotic platform with a virtual representation respecting two important constraints which are the accuracy and the safety. The trajectories design is performed on a virtual platform and our work will eventually provide all necessary implementation and exploitation tools of trajectories without any real loss of performance.

The realized system should bring to the automotive industry a tool allowing the manipulation of real robot via a GUI and assuring a perfect portability with instant data from the virtual scene to the robotics workshop. Moreover, this tool contains a new mode of human-robot interaction relating operators and industrial robotic environments. Finally, this interface guarantee an assistance to teach industrial robots using mobile and flexible modules that can be declined in several types or forms to ensure greater portability.

3 Our approach

To take advantage of the two teaching approaches presented previously and overcome drawbacks of each method, we propose a new technique mixing the two techniques, namely: the manual teaching and the CAD. The originality is to perform the robot trajectory teaching using an assistance tool based on an AR system. The idea is to avoid, initially, the use of the real workpiece. Indeed, trajectory teaching will be carried out on a virtual workpiece using the real robot. To assist the operator, a virtual waypoints tool will be used to force the robot to go on the desired impact point. The virtual guides will allow to guide the operator by reaching different impact points to improve accuracy and quality of work.

This system allows to create the trajectory on the workpiece being developed without using the real prototype. The robot control is carried out remotely via network communications to ensure the operator safety. The environmental constraints of the robot such as congestion and fittings movement are addressed directly with camera feedback. The solution we propose enables avoidance of approximate teaching, a virtual registered 3D trajectory is superimposed on the real path that the robot should cover. On the other hand, we improve human safety by avoiding collision risks since the operator is carrying out his task using a real-time video feedback. Moreover, an important prerequisite of our system is the positioning accuracy. Since we use an AR tracking system, the accuracy of the application depends on pose estimation process. Therefore, a comparison study of several methods is a prerequisite of our application to drive and implement the most reliable one to locate the virtual workpiece in the real environment respecting the accuracy margin of robotic tasks in automotive industry (figure 1).

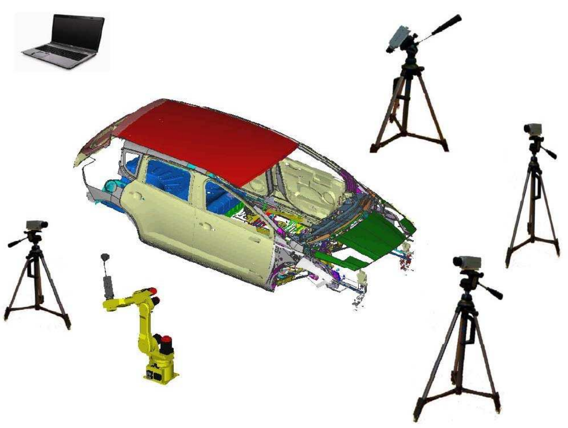

Figure 1 System overview.

4 System description

The main goal of our system is to create trajectories on virtual workpieces using a real robot and computer generated models. The designed system is an assistant tool for robot trajectories teaching on virtual elements. It consists in a virtual platform interacting with a robot. This platform is able to reproduce all main features necessary to program robot routine tasks.

These features are available to the user via a computer and a robot control interface, the communication between the virtual platform and the robotics workshop is realized via Ethernet.

The operator controls the robot via a network communication and has a real-time feedback of its position. Four cameras are used: 3 cameras to visualize the robot and the workpiece under different viewing angles and the fourth camera serves to supervise the whole system.

The material used in this project consists of a robot and a vision system. The robot is the LRMate 200iB (figure 2), an industrial 6 DOF robot from Fanuc Robotics having the following specifications:

- Axes: 6
- Payload: $5kg$
- H-Reach: $700mm$
- Repeatability: $\pm 0.04mm$
- Robot Mass: $45kg$
- Structure: *Articulated*

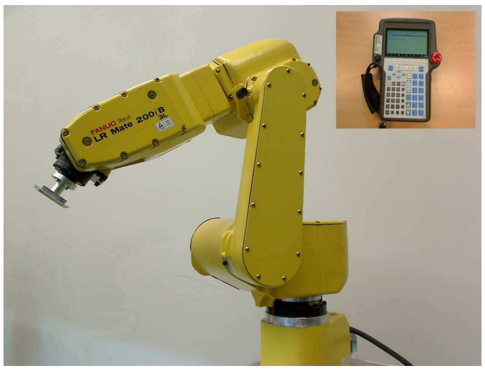

Figure 2 LR Mate 200iB Robot.

The vision system is composed of 4 PixeLINK PL-B762F cameras to visualize the robot and the workpiece from different views (figure 3). These are industrial cameras for machine vision applications containing CCD sensors coupled with Firewire digital bus technologies. We used $4.5mm$ focal length objectives for a wide vision range. The effect of this short focal length is to allow a broader frame of close objects and have a global view of the system since the robot needs a large workspace for operating. The camera specifications are:

- Resolution: 752 × 480
- Sensor type: CMOS
- FPS at full resolution: 60
- Interface: Firewire

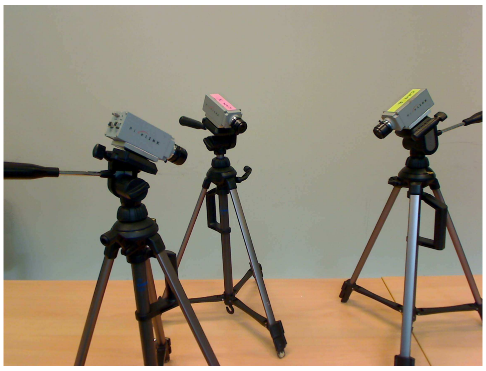
Figure 3 PixeLINK cameras.

The camera set is positioned around the robot to have 3 points of view and visualize the virtual workpiece from 3 sides.Tripods are used to hold cameras, ensure a rigid fastening and allow easy and accurate rotations. The robot could be oriented following several articulations around its 6 axes using the Teach Pendant which controls positions, orientations, speed and displacement step (figure 4).

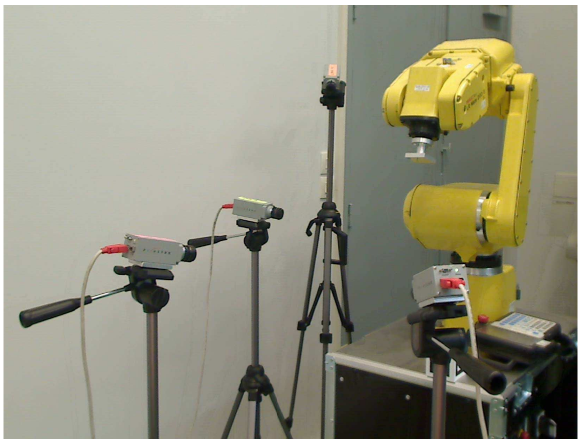
Figure 4 The global system.

5 Implementation

We developed a Graphic User Interface (GUI) to interact with materials and to control the robot. The GUI contains 3 parts: the first one allows the interface control, this phase involves the creation of application controls, the association of functionalities to the components and the management of the client-server connection between the GUI and the robot. The second part is to integrate the vision module to display real-time video stream from cameras. This module includes the AR application allowing the visualization and localization of virtual objects. The third part of the GUI is the implementation and the integration of the Reverse Tool Control Point (RTCP) mode which uses the robot data positioning to compute the camera pose.

The conceived GUI allows the network management, the robot control and the cameras views display.

5.1 Robot control

The robot can operate according to the Cartesian mode in which it moves along $(X, Y, Z, Roll, Pitch, Yaw)$ or Joint mode where it moves following its 6 rotations axes $(J_1, J_2, J_3, J_4, J_5, J_6)$.

The GUI and the robot communicate via data frames: controls and positions frames.

In Cartesian mode, 3 steps are defined:

- δ: translation step in millimeters.
- θ: angle step in degrees.
- σ: percentage of the maximum motion speed of the robot.

To ensure safety, we have decided to limit the maximum translation and rotation step of the robot to $100mm$ and $45°$ respectively. Concerning the robot speed, we limited it to 20% ($2000mm/s$). The data frame has the following syntax: $[\delta X, \delta Y, \delta Z, \theta Roll, \theta Pitch, \theta Yaw, 0G, \sigma S]$

In Cartesian mode, the control frame always begins with $'['$ and finishes with $']'$. X, Y, Z represent translations and Roll, Pitch, Yaw are the rotation angles. G denotes the opening control of the robot's gripper. This data is not used in our case, therefore, $G = 0$. S denotes the robot speed.

The joint mode transcribe the frame control as following: $(\theta A, \theta B, \theta C, \theta D, \theta E, \theta F, 0G, \sigma S)$. The control frame of the Joint mode always begins with $'\{'$ and finishes with $'\}'$. A, B, C, D, E, F denotes respectively the rotation axes of the robot around $J_1, J_2, J_3, J_4, J_5, J_6$.

5.2 The vision system

Four cameras have been used for the application, 3 cameras are positioned face to the robot and oriented 90° from each other to get different virtual workpiece view while the fourth camera supervises the whole system and provides a scene overview (figure 5).

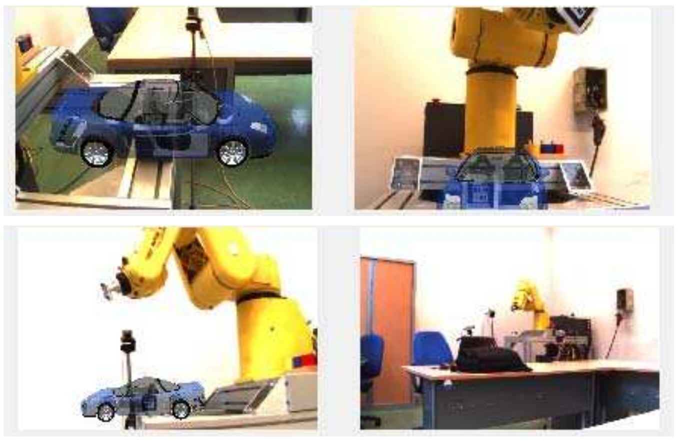

Figure 5 Video feedback from cameras.

The tracking process in AR is composed of two steps: object recognition and pose estimation. For the identification part, we designed specific targets using the most popular algorithm in AR applications, the ARToolKit [13]. However, for pose estimation, we compared 3 algorithms to study their performance and derive the one which satisfies the application accuracy requirement. ARToolKit is a marker system used in AR systems. Thanks to its robustness performance, it is used in a lot of AR and vision applications. ARToolKit includes several models of two-dimensional fiducial markers. It allows to find markers, to recognize and identify them. However, its performance in markers detection should be improved. In fact, often, it happens that markers are confused with each other or they are detected by error in foreground. ARToolKit marker is black border square surrounding a model which is compared to other pre-recorded models in ARToolKit matching template database (figure 6).

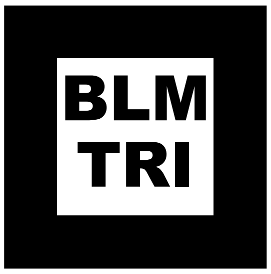

Figure 6 ARToolKit marker.

The identification process in ARToolKit library is composed of several steps, the first is the image binarization using an appropriate threshold. Then a search of connected components is performed to determine connected regions in the image. The edges and corners are then extracted from the image and finally, the 2D/3D points are matched and used afterwards for pose estimation process.

5.3 *Pose estimation*

The ICP algorithm

ARToolKit uses the Iterative Closest Point (ICP) algorithm to estimate position and orientation of the target. The ICP algorithm is usually used to register two given point sets in a common coordinate system. The algorithm calculates iteratively the registration. In each iteration step, the algorithm selects the closest points as correspondences and calculates the rotation and the translation (R, T) by minimizing the equation:

$$E\left(R,T\right) = \sum_{i=1}^{N_m}\sum_{j=1}^{N_d} w_{i,j}\left\|m_i - \left(Rd_j + T\right)\right\|^2 \tag{1}$$

Where N_m and N_d, are the number of points in the model set m, and the data set d, and w_{ij} are the weights for a point match.

The ICP algorithm is widely used for the registration of geometric data. One of its main drawback is its time complexity $O(N^2)$, which implies long processing time, especially when using high resolution data.

The other practical difficulty of the ICP algorithm is the accuracy of the search for correspondence points which highly affects the estimation of the transformation parameters, the output of the first step has a major impact over the following stages and strongly affects the overall performance of the algorithm. This step strongly depends upon both the selection of the points of the two surfaces, and the method used for finding the correspondence of the selected points. Since, the algorithm is iterative, the convergence of the algorithm depends on the error criterion and number of iterations. This has an impact on both accuracy and execution time.

For these reasons, we implemented two other pose estimation algorithms to make a comparative study and retain the most accurate method for implementation. Indeed, the accuracy is the most relevant criterion for our system since it consists in performing trajectories on virtual workpieces that should be reproduced on real manufactured pieces later.

The Zhang analytical pose estimator

The second algorithm that we implemented is the analytical pose estimator based on Zhang technique [24].

The technique requires 2D/3D matching points to solve the transformation allowing the determination of pose parameters.

The relationship between a 3D point P and its image projection p is given by:

$$s\tilde{p} = A\left[R\,T\right]\tilde{P} \tag{2}$$

where $\tilde{p} = (u, v, 1)^t$ denotes a 2D point and $\tilde{P} = (X, Y, Z, 1)^t$ a 3D point. A is the camera intrinsic matrix, R is the rotation matrix, T represents the translation vector, and finally, s is an arbitrary scale factor.

The technique assumes that the model is plane on $Z = 0$ of the world coordinate system. Let's denote the i^{th} column of the rotation matrix R by r_i. From equation 2, we have:

$$s \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = A \begin{pmatrix} r_1 \, r_2 \, r_3 \, T \end{pmatrix} \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} = A \begin{pmatrix} r_1 \, r_2 \, T \end{pmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{3}$$

Given an image of the model plane, an homography can be estimated. We denote it by $H = (h_1 \, h_2 \, h_3)$ which is identified to $H = A(r_1 \, r_2 \, T)$. Once A is known, the pose parameters for each image is readily computed as follows:

$$\begin{array}{l} r_1 = \lambda A^{-1} h_1 \\ r_2 = \lambda A^{-1} h_2 \\ r_3 = r_1 \times r_3 \\ T = \lambda A^{-1} h_3 \end{array} \tag{4}$$

The Orthogonal Iteration algorithm

In this method, the pose estimation is formulated as error metric minimization based on collinearity in object space. Using object space collinearity error (figure 7), an iterative algorithm is derived to compute orthogonal rotation matrices [15].

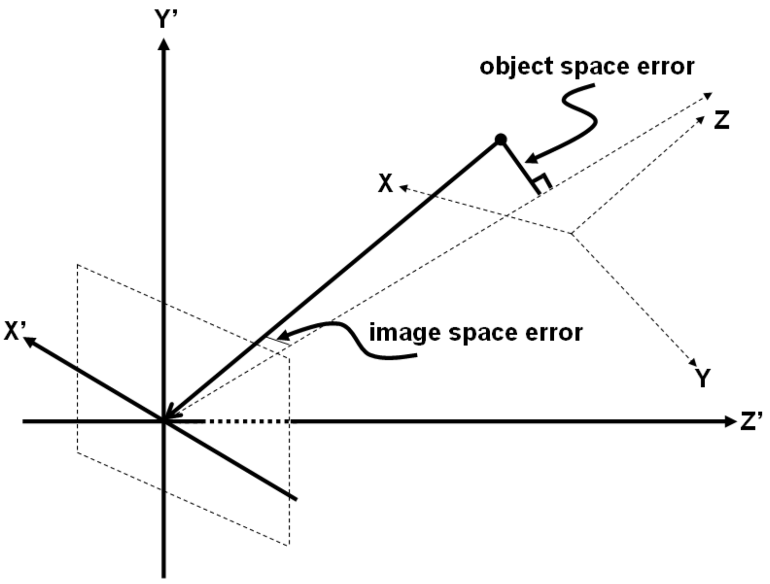

Figure 7 Object-space and image-space collinearity errors.

The mapping from 3D reference points to 2D image coordinates is formalized as follows: given a set of noncollinear 3D coordinates of reference points $P_i = (x_i, y_i, z_i)^t$, where: $i = 1...n, n \geq 3$, expressed in an object-centered reference frame, the corresponding camera-space coordinates $q_i = ({x_i}', {y_i}', {z_i}')^t$, are related by a rigid transformation as: $q_i = RP_i + T$, where R and T are respectively the rotation matrix and the translation vector. The reference points P_i are projected to the image plane. Let the image point $p_i = (u_i, v_i)^t$, be the projection of P_i on the normalized image plane. Under the idealized pinhole imaging model, p_i, q_i, and the center of projection are collinear. This fact is expressed by the following equation:

$$u_i = \frac{r_1^t P_i + t_x}{r_3^t P_i + t_z} \tag{5}$$

$$v_i = \frac{r_2^t P_i + t_y}{r_3^t P_i + t_z} \tag{6}$$

and

$$p_i = \frac{1}{r_3^t P_i + t_z} \left(RP_i + T \right) \tag{7}$$

The OI algorithm allows to dynamically determine the external camera parameters using 2D-3D matchings established by the 2D fiducials tracking algorithm from the current video image. The OI algorithm computes first the object-space collinearity error vector [15]:

$$e_i = \left(I - \hat{V}_i \right) \left(RP_i + T \right) \tag{8}$$

where $\hat{V}_i$ is the observed line of sight projection matrix defined by:

$$\hat{V}_i = \frac{\hat{p}_i \hat{p}_i^t}{\hat{p}_i^t \hat{p}_i} \tag{9}$$

then, a minimization of squared error is performed:

$$E\left(R, T\right) = \sum_{i=0}^{n} \left\| e_i \right\|^2 = \sum_{i=0}^{n} \left\| \left(I - \hat{V}_i \right) \left(RP_i + T \right) \right\|^2 \tag{10}$$

The OI algorithm converges to an optimum for any set of observed points and any starting point.

6 Experimental results

We present now, experimental results and a detailed evaluation of different localization methods that we presented before. A comparison between these methods is performed in order to determine the most accurate one. We compared 3 pose estimation algorithms which are the analytical algorithm of Zhang, the ICP algorithm and the OI algorithm. The comparison between these algorithms is carried out according to the following criteria:

- Execution time

- Reconstruction error: measures the pixellic difference between feature points of the detected target in the image and the 3D target model projection using the computed pose parameters.
- Generalization error: consists on projecting the target which was not used for pose computation on the image plan and measure the variation in pixels between the projected points of the 3D models and the corresponding target feature points detected in the image.
- Real intra-targets distance estimation: it is the difference between the estimated distance computed by the pose algorithms and the real distance measured between two targets.

The experimental study was realized using a PC with the following material configuration:

- Intel Core 2 Quad @ 2.4GHz.
- 3GB RAM.

The camera is calibrated and the intrinsic parameters are given in table 1.

Table 1 Intrinsic parameters of the PixeLINK camera used in experiments.

Image size (px)			752×480
Projection parameters		**Distortion parameters**	
Scale factors		Radial distortion coefficients	
$\alpha_u(px)$	928.48	k_1	-0.2279
$\alpha_v(px)$	926.47	k_2	0.1479
Optical center projection		Tangential distortion coefficients	
$u_0(px)$	339	p_1	-0.0007985
$v_0(px)$	215	p_2	0.0006245

The first experiment is a qualitative test that consists in performing an arbitrary motion of the target around the camera. The 3 pose estimation algorithms compute the rotation and the translation of the target reference frame according to the camera reference frame. From figure 8, we notice that the translation of the 3 algorithms has roughly the same aspect except in Z direction where we observe a quite shift between graphics. This is due to the fact that the camera is a monocular sensor and we could not obtain a real depth estimation with this kind of device.

On the other hand rotation parameters present some shifts between the 3 algorithms (figure 9). The rotation is expressed with quaternions because of their compactness and avoidance of discontinuous jumps. We can notice that q_x which is the angle estimation, is quite different from each pose estimator and several peaks appear on other rotation components, especially the analytical algorithm which is not appropriate to estimate such parameters. Indeed, this algorithm computes a direct transformation without any minimization of error criteria.

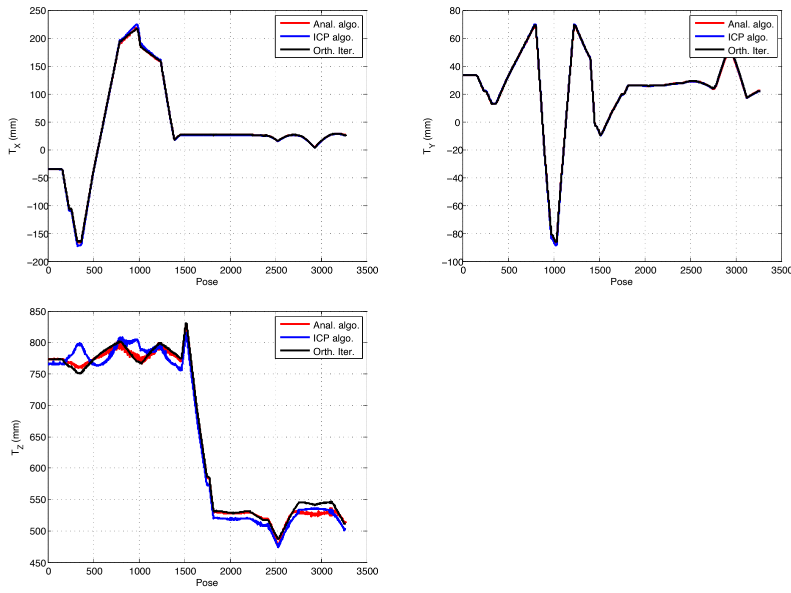

Figure 8 Estimation of translation with the 3 algorithms.

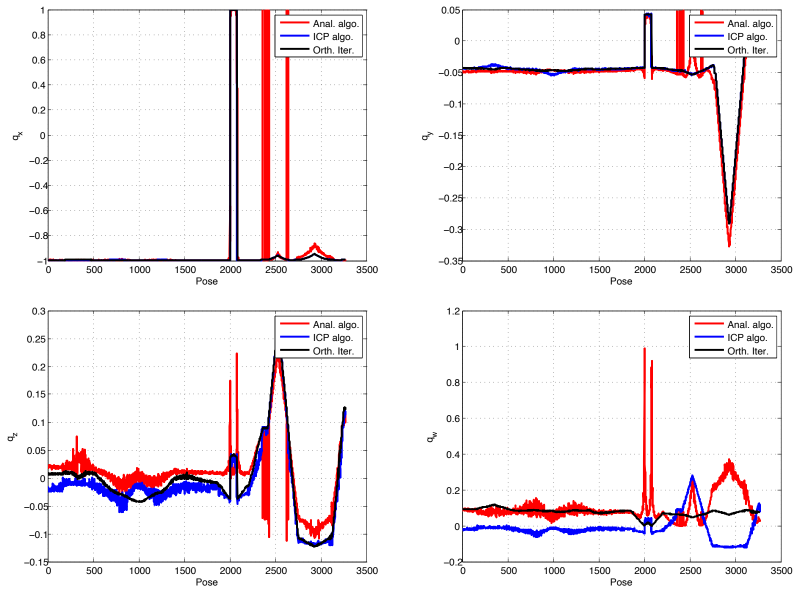

Figure 9 Estimation of rotation with the 3 algorithms.

6.1 Execution time

We were interested after to execution time of different algorithms. We estimated 3247 pose for the 3 pose estimators. The results showed that the analytical algorithm is the fastest method with $0.0986ms$ for one pose estimation, the ICP algorithm makes $0.6453ms$ to estimate the same pose, and finally, $1.6255ms$ are necessary for the OI to determine pose parameters. So, in term of computation time, we can say that the analytical algorithm is faster in regard to other algorithms which are also real time efficient and this could not compromise the visual rendering.

6.2 Reconstruction error

In this experimentation, the camera is moved around the target object, the 3 algorithms estimate the pose parameters and we evaluate the reconstruction error in the image. The 3 algorithms computed 3270 poses, the error is estimated by re-projecting the object model on the image. For each pose computation, we re-project the target model on the image and we measure the deviation between real target corners and the projected corners. In table 2, we notice that the analytical algorithm is the most stable and accurate method comparing to the other algorithms. From figure 10, we can see that the analytical and the OI methods present the lowest reconstruction error, the two algorithms are accurate and stable. The reconstruction error is important for the ICP, the algorithm doesn't converge to the optimal solution.

Table 2 Results on different experiments performed for reconstruction error.

Algorithm	**Anal. algo.**	**ICP**	**OI**
Reconst. error (px)	0.0048	1.8293	0.2232
Variance (px^2)	$4.65\ 10^{-6}$	0.1780	0.0143
Standard deviation (px)	0.0022	0.4220	0.1196

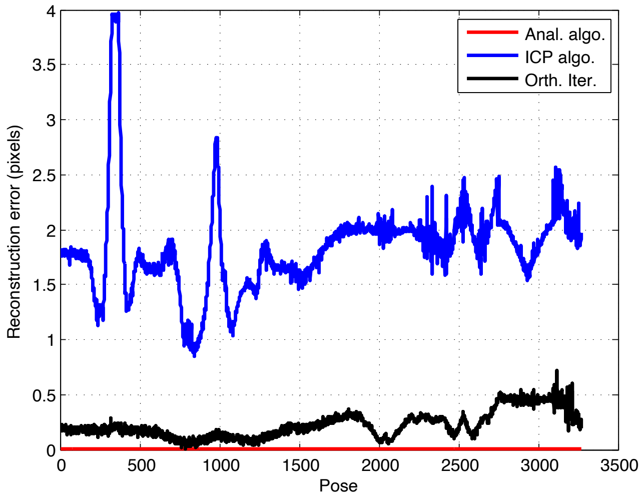

Figure 10 Reconstruction error.

6.3 Generalization error

To determine the generalization error, we carried out a series of 5245 pose to compute this error. We used a paper in which we printed two square targets with $5cm$ side. The first target is used to compute pose parameters and the second target is used to compute the generalization error. This generalization error is computed by re-projecting the object model which didn't serve to estimate the pose. The obtained results on generalization error are represented in figure 11, from the curves we notice that the overall error behavior of the OI algorithm is stable and don't present jitter in images comparing to the other algorithms. Table 3 shows that the OI presents the best performance in term of generalization error, the numerical results proved the effectiveness of this algorithm to extend overlaying on other scene elements using a single target pose computation.

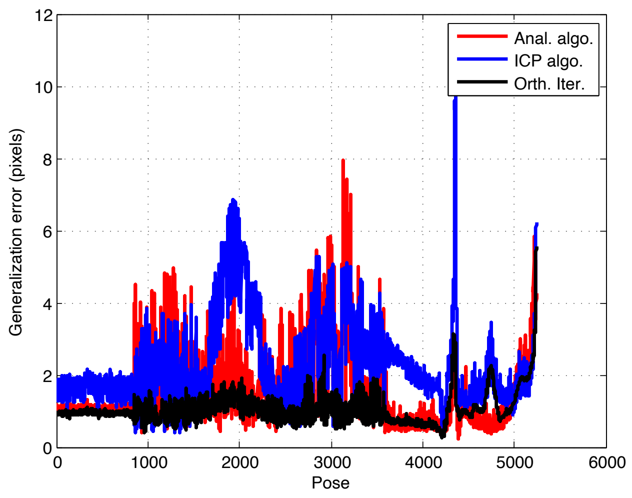

Figure 11 Generalization error.

Table 3 Results on different experiments performed for generalization error.

Algorithm	**Anal. algo.**	**ICP**	**OI**
Gener. error (px)	1.5935	2.3127	1.0652
Variance (px^2)	1.2123	1.6322	0.1953
Standard deviation (px)	1.1010	1.2776	0.4419

6.4 Real distance estimation

In order to evaluate a real distance between two targets and compute distance estimation errors with the different algorithms, we attached the targets to the robot tool and we performed several displacements. The robot generates motions and the algorithms compute position and we recorded these poses to compare them with the real distance measured between targets. We sample the robot displacement space in order to compute the corresponding pose with the different pose estimators. We have 2754 robot positions for which each algorithm estimates the pose parameters and computes the distance between the optical center of the camera and the two targets, once we have the position of the two targets according to the camera, we deduce the distance between these two targets with geometrical calculation.

The results are illustrated in figure 12, the graphics represent the real measured distance between targets and the position estimated by the pose algorithms. Moreover, we computed the mean error, variance and standard deviation of the pose estimation methods. From table 4, we notice that the ICP method presents a considerable mean error compared to other methods, its variance and standard deviation are also important. The OI presents the best performances, unlike the analytical algorithm which presents a quite large variance around its mean error. Finally, this evaluation determines the most important performance criterion which is the localization accuracy representing the fundamental requirement for building a reliable waypoints tracking on virtual worpieces.

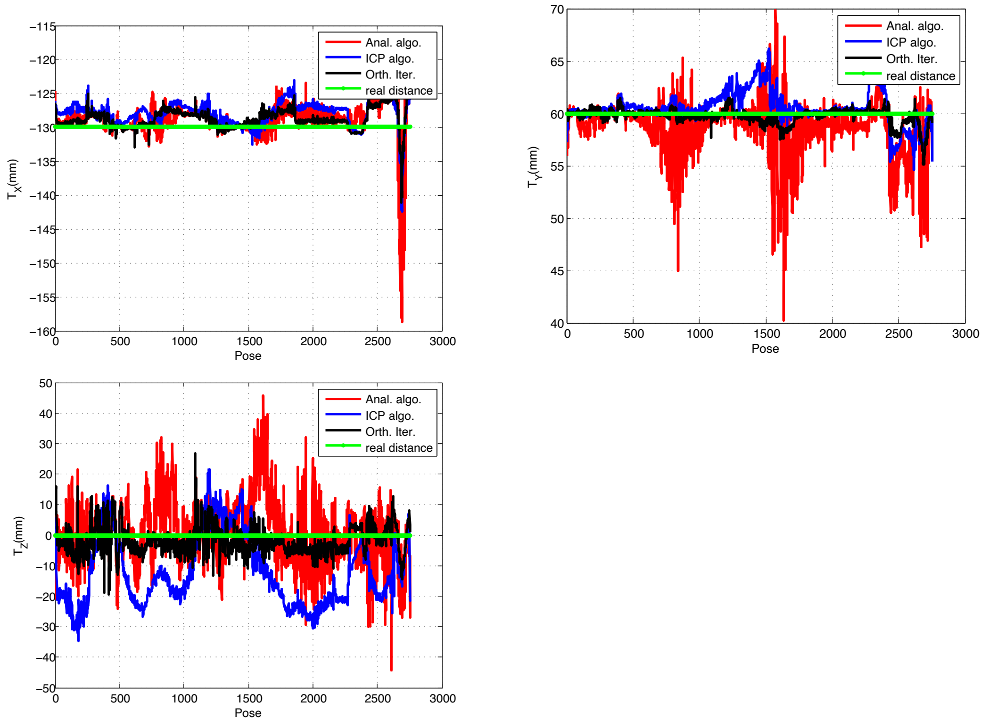

Figure 12 Real distance estimation.

Table 4 Results on different experiments performed on distance estimation.

Algorithm		**Anal. algo.**	**ICP**	**OI**
Mean error (mm)	T_x	1.8172	2.4004	1.4530
	T_y	1.9546	1.0969	0.4857
	T_z	7.9597	14.7197	4.1308
Variance (mm^2)	T_x	7.8781	3.6291	2.1409
	T_y	7.2434	2.3495	0.5441
	T_z	107.2061	135.8691	21.2878
Standard deviation (mm)	T_x	2.8068	1.9050	1.4632
	T_y	2.6913	1.5328	0.7376
	T_z	10.3540	11.6563	4.6139

6.5 Comparison between camera pose and robot pose

This experiment consists in computing the position and the orientation of the target reference frame according to the camera reference frame using a pose estimator algorithm and the positioning data returned by robot.

The objective of this experiment is to check the accuracy of the ICP and OI algorithms comparing to the robot pose. In this experiment, the target is attached to the robot tool, the pose estimator computes translation and orientation and the robot returns also these parameters. Indeed, the robot gives the tool pose according to the robot basis, a calibration step is required to determine the transformation between the camera reference frame and the robot basis.

We carried out the experiment with 2785 robot positions and we computed at the same time the transformation relating the robot tool to the camera using the ICP and the OI algorithms. Figure 13 shows the translation results, the robot serves as reference and the pose estimators compute the same transformation. We notice that in X and Y directions the translation is well estimated with both ICP and OI, however, in Z direction, the OI is more accurate than the ICP. In table 5 we have the numerical results of pose estimation, we see clearly that the OI brings more accuracy and robustness to pose estimation. The orientation is quite stable with the OI algorithm and presents some wavering with the ICP. Nevertheless, the general aspect of rotation curves is stable and approximates the reference rotations (figure 14).

Table 5 Camera translation results compared to robot position.

Mean Error Orth. Iter./Robot pose (mm)		
T_x	T_y	T_z
0.7598	0.8123	5.6531
Mean Error ICP/Robot pose (mm)		
T_x	T_y	T_z
0.8925	0.8079	8.4469

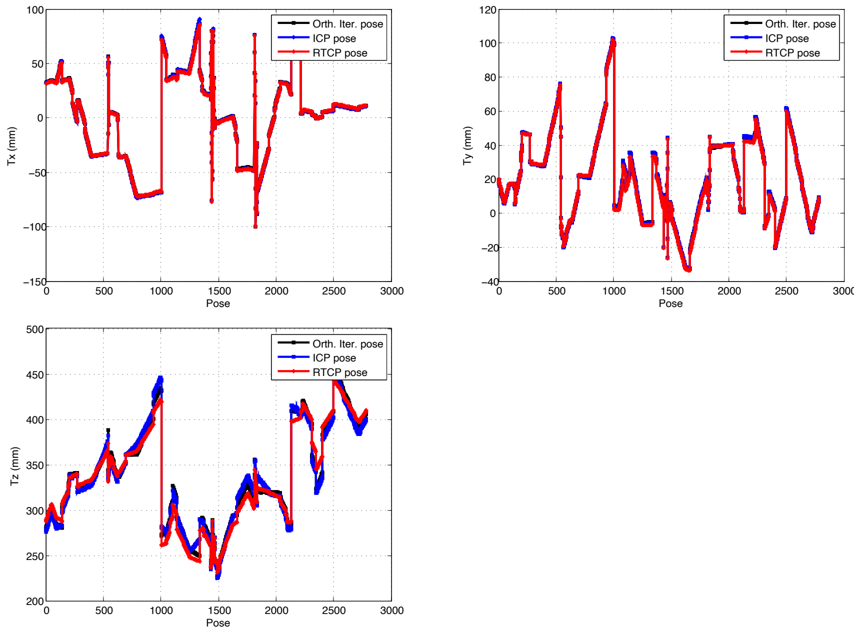

Figure 13 Comparison between the camera estimated translation and the robot positions.

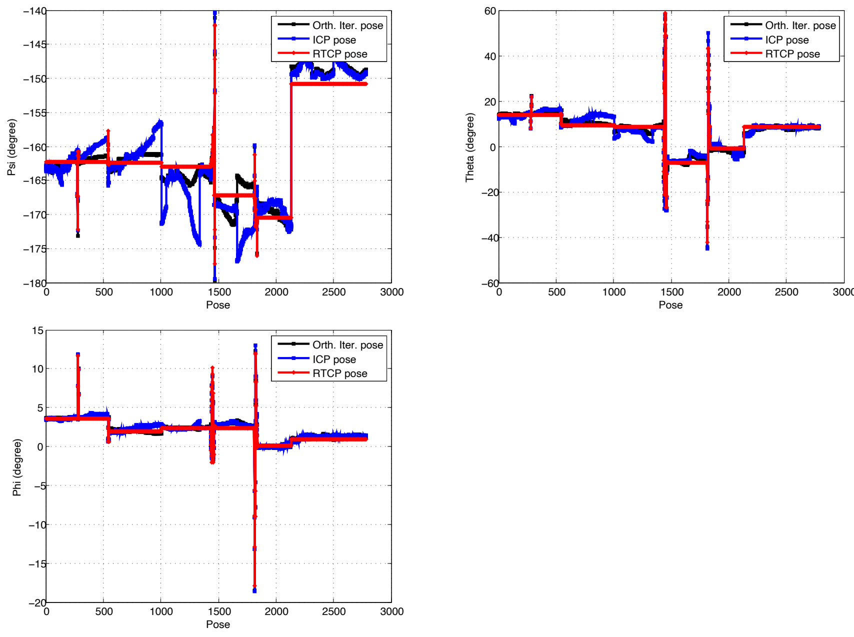

Figure 14 Comparison between the camera estimated rotation and the robot rotations.

7 Discussion

In this work, we presented a system intended to teach robot trajectories in automotive industry. The major issue of our application is ensuring positioning accuracy of virtual workpieces. For this purpose, we studied the accuracy and robustness of several pose estimators in order to compare performances of each method and draw the most reliable and pertinent localization method.

We compared the performances of 3 pose estimation algorithms. We evaluated these methods using an experimental protocol to compute several error sources and estimate real distances. We used two iterative methods depending on nonlinear optimization and an analytical method based on direct computation of parameters.

The main accomplishments of this work are:

- Realization of an original system dedicated to teach industrial robots.
- Experimentation of the system in overlaying and localization tasks using robot positioning data.
- Comparison of different pose estimation methods in term of execution time, errors and real distance estimation.

The main experimental test of our system concerns pose algorithms evaluation since it is the most important factor to carry out accurate trajectories on virtual workpieces. These trajectories represent a waypoints for displacement and should be recorded on the robot program to be recurred thereafter on real vehicle pieces. The comparison of several pose estimator was primary requirement of our application to draw the effective method presenting the less error sensitivity.

Previous published papers on vision-based pose estimation used direct or iterative methods and some authors were interested in comparison and evaluation of these methods. DeMenthon and Davis [7] have compared several approximate methods for the perspective 3 point problem to solve the pose estimation parameters. A synthesis work was realized in [1], the authors developed a fast and accurate analytical pose estimation algorithm for a limited numbers of points or lines. Their method was tested and compared to linear algorithms and also some iterative methods.

In table 6, we compare different pose estimation methods, where we precise the year, the nature of the algorithm and the condition of application.

Table 6 Summarization of pose estimation methods.

Method	**Year**	**Type**	**Application condition**
Dhome et al [8]	1989	Analytical	3 lines
Dementhon and Davis [7]	1992	Analytical	3 points
POSIT	1995	Iterative	4 non coplanar points
OI [15]	2000	Iterative	3 points
EKF [5]	2002	Iterative	3 points
Ansar and Daniilidis [1]	2003	Analytical	4 points
Hybrid EKF [17]	2007	Analytical and iterative	4 coplanar points

We quantitatively analyzed the tracking and localization errors of 3 algorithms to locate drawbacks of each method and enhance accuracy and robustness of our system by implementing the most appropriated method. The distance estimation was the major issue that our study addressed to contribute, particularly, to the improvement of depth computation.

Indeed, the two kinds of algorithms have advantages and shortcomings. Iterative methods are accurate but suffer from computation expense due to bad initialization and local minima problems. On the other side, the analytical methods are fast but their major disadvantage is the lack of accuracy. In our case and since the accuracy is the major constraint that the system should respect, we used an iterative method based on error minimization criterion. The OI proved its effectiveness, particularly, in distance estimation, the results of the experimental protocol performed during our study showed the real advantage of implementing this method for the pose estimation process.

8 Conclusion

In this paper, we presented an AR system intended to trajectory teaching in automotive industry. The application requires robust and accurate positioning of virtual object, however, this accuracy depends on localization algorithms used to compute the camera pose. For this purpose, we performed a comparative study of 3 pose estimators, the target is identified using the correlation technique of ARToolKit, then, we implemented 3 pose estimation methods: the analytical pose estimators of Zhang, the ICP and the OI algorithm. The analytical method computes rotation and translation parameters using a direct computation of solution, the ICP is an iterative method used in ARToolKit library and OI is also an iterative technique which formulates an error metric minimization based on collinearity in object space.

We evaluated the performances of our localization system by comparing these 3 algorithms. This study related to the following performances criteria: execution time, reconstruction error, generalization error and real distance estimation. The experimentation tests to estimate errors were realized using the robot for motion generation and as positioning reference system since it provides localization of its tool according to a known world reference frame. The obtained results for OI algorithm were efficient and robust and proved that this method provides interesting solutions for camera localization using AR targets.

Obviously, for industrial application in which the localization error must be in the order of millimeter, compensation techniques should be developed to compensate the error rate perceived in Z direction. Another solution consists in using the camera set data to proceed to depth rectification since the positions of cameras are known. Indeed, the fusion of data pose of the whole camera system could fit the depth estimation by exploiting geometrical relationships relating the coordinate reference frames.

References

[1] A. Ansar and K. Daniilidis. Linear pose estimation from points or lines. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):578–589, 2003.

[2] H. Araujo, R. Carceroni, and C. Brown. *A Fully Projective Formulation for Lowe's Tracking Algorithm*. Technical Report 641, University of Rochester, USA, 1996.

[3] G. Biegelbauer, A. Pichler, M. Vincze, C. Nielsen, H. Andersen, and K. Haeusler. The inverse approach of flexpaint. *IEEE Robotics & Automation Magazine*, 12(3):24–34, 2005.

[4] R. Bischoff and J. Kurth. Concepts, tools and devices for facilitating interaction with industrial robots through augmented reality. In *ISMAR Workshop on Industrial Augmented Reality*, Santa Barbara, CA, USA, October 22 2006.

[5] L. Chai, W. A. Hoff, and T. Vincent. Three-dimensional motion and structure estimation using inertial sensors and computer vision for augmented reality. In *Presence: Teleoperators and Virtual Environments*, volume 11, pages 474–492, Cambridge, MA, USA, 2002.

[6] J. W. S. Chong, Nee, Y. Andrew, K. Youcef-Toumi, and S. K. Ong. An application of augmented reality (ar) in the teaching of an arc welding robot. *Innovation in Manufacturing Systems and Technology (IMST)*, 2005.

[7] D. DeMenthon and L. S. Davis. Exact and approximate solutions of the perspective-three-point problem. *In IEEE Trans. Pattern Anal. Mach. Intell.*, 14(11):1100–1105, 1992.

[8] M. Dhome, M. Richetin, J. T. Lapreste, and G. Rives. Determination of the attitude of 3d objects from a single perspective view. *In IEEE Trans. Pattern Anal. Mach. Intell.*, 11(12):1265–1278, 1989.

[9] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[10] W. Forstner. Reliability analysis of parameter estimation in linear models with applications to mensuration problems in computer vision. *Computer Vision, Graphics and Image Processing*, 40:273–310, 1987.

[11] R. M. Haralick, K. O. C. Lee, and M. Nolle. Analysis and solutions of the three point perspective pose estimation problem. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 592–598, Maui, Hawaii, 1991.

[12] B. K. B. Horn, H. Hilden, and S. Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *In Journal of the Optical Society of America A*, 5:1127–1135, July 1988.

[13] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top ar environment. In *Proceedings of the International Symposium on Augmented Reality (ISAR'2000)*, pages 111–119, Munich, Germany, October 2000.

[14] D. G. Lowe. Three-dimensional object recognition from single two-dimensional image. *In Artificial Intelligence*, 31:355–395, 1987.

[15] C. P. Lu, G. D. Hager, and E. Mjolsness. Fast and globally convergent pose estimation from video images. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):610–622, 2000.

[16] M. Maidi, F. Ababsa, and M. Mallem. Active contours motion based on optical flow for tracking in augmented reality. In *8th International Conference on Virtual Reality (VRIC'06)*, pages 215–222, Laval, France, 2006.

[17] M. Maidi, J.-Y. Didier, F. Ababsa, and M. Mallem. A performance study for camera pose estimation using visual marker based tracking. *Machine Vision and Applications, IAPR International Journal, Springer*, 2008.

[18] H. P. Moravec. Towards automatic visual obstacle avoidance. In *Proc. 5th Int. Joint Conf. Artificial Intell.*, volume 2, page 584, Cambridge, Massachusetts, USA, August 1977.

[19] S. K. Ong, J. W. S. Chong, and A. Y. C. Nee. Methodologies for immersive robot programming in an augmented reality environment. In *GRAPHITE '06: Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, pages 237–244, New York, NY, USA, 2006. ACM.

[20] T. Pettersen, J. Pretlove, C. Skourup, T. Engedal, and T. Lkstad. Augmented reality for programming industrial robots. In *ISMAR '03: Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, page 319, Washington, DC, USA, 2003. IEEE Computer Society.

[21] L. Quan and Z. D. Lan. Linear n-point camera pose determination. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):774–780, 1999.

[22] K. Satoh, K. Takemoto, S. Uchiyama, and H. Yamamoto. A registration evaluation system using an industrial robot. In *ISMAR '06: Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 79–87, Washington, DC, USA, 2006. IEEE Computer Society.

[23] N. Shimizu, M. Sugimoto, D. Sekiguchi, S. Hasegawa, and M. Inami. Mixed reality robotic user interface: virtual kinematics to enhance robot motion. In *ACE '08: Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, pages 166–169, New York, NY, USA, 2008. ACM.

[24] Z. Zhang. A flexible new technique for camera calibration. Technical Report MSR-TR-98-71, Microsoft Research, Microsoft Corporation, Redmond, WA, USA, 1998.