2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Visual Servoing by Optimization of a 2D/3D Hybrid Objective Function

A. H. Abdul Hafez [1,2] and C.V. Jawahar [1]

Abstract— **In this paper, we present a new hybrid visual servoing algorithm for robot arm positioning task. Hybrid methods in visual servoing partially combine the 2D and 3D visual information to improve the performance of the traditional image-based and position-based visual servoing. Our algorithm is superior to the state of the art hybrid methods. The objective function has been designed to include the full 2D and 3D information available either from the CAD model or from the partial reconstruction process by decomposing the homography matrix between two views. Here, each of 2D and 3D error functions is used to control the six degrees of freedom. We call this method 5D visual servoing. The positioning task has been formulated as a minimization problem. Gradient decent as a first order approximation and Gauss-Newton as a second order approximation are considered in this paper. Simulation results show that these two methods provide an efficient solution to the camera retreat and features visibility problems. The camera trajectory in the Cartesian space is also shown to be satisfactory.**

I. INTRODUCTION

Visual servoing has become an attractive area of research, and has recently received considerable amount of attention [1], [2], [3], [4], [5]. Visual servoing schemes use one or more cameras along with computer vision algorithms to control the position of a robot arm or mobile robot with respect to an object or a set of features of the object to be manipulated. It is used in a wide range of applications such as robot navigation, lane tracking by vehicles, and industrial manipulation.

The essence of visual servoing is to move the concerned object from the current pose to a desired pose given by current and desired images. This is essentially obtained by minimization of a cost function. Visual features are extracted from the two images and used to formulate a function of the error between the current pose and the desired one. The role of the minimization process is to regulate this error function to zero. Image features can be used directly in the definition of the error function. This leads to a formulation of the 2D error function that is minimized in the image space. These image features may also be used to formulate an error function in the pose space.

In image-based visual servoing, the input is computed in the 2-D image space, and called image-based visual servoing. This method is robust to robot and camera calibration errors. However, image-based visual servoing is known to be locally stable with a stability region that is difficult to determine. This is in addition to the need of the depth estimates of the image features. It was shown in [6] that the stability domain with respect to depth estimates is not very wide. On the other hand, in position-based visual servoing, the input is computed in the 3D Cartesian space. This method is also called 3D visual servoing. Assuming that the model of the target is perfectly known, the pose of the target with respect to the camera is estimated from the image features. The pose is recovered at each iteration and the velocity command is computed proportional to the pose error between the current and the desired states.

The main drawback of 3D visual servoing is the lack of control in the image space. This implies that the object may get out of the camera field of view. In contrast, 2D visual servoing does not have any control in the Cartesian space and the camera trajectory is not predictable. A satisfactory control scheme that avoids these drawbacks [2], [7] is to use both kinds of information *i.e.*, 2D and 3D. The 2D information is obtained from the image space while the 3D information is obtained from the pose estimation process of the object or from partial pose estimate computed from the current and desired images. A 6-D error vector consisting of 3D information about the rotation and 2D information about the translation had been employed. For this reason, this control scheme is called 2 1/2D visual servoing [2].

We proposed in [8] a hybrid (2D-3D) cost function that contains information from the 2D image space and the 3D pose space. In this paper, we apply this objective function minimization to the visual servoing process. The closest work to ours is the error function that was proposed in the visual servoing literature [2], [7].

In [2], the error function was defined as a 6-vector. The 3-vector that contains the 2D visual information from the image space is used to recover the position of the camera, where the 3-vector that contains the 3D visual information from the pose space is used to recover the orientation of the camera. In contrast, our method uses each of the image space information and pose space information to recover the full camera pose *i.e.*, both position and orientation. Both 2D and 3D errors are concatenated into a 12-vector and minimized together. In such a minimization method, the minimization process searches for a least squares solution that minimizes the 2D error from the image space and the 3D error from the pose space simultaneously.

[1] Center for Visual Information technology, International Institute of information Technology, Gachibowli, Hyderabad-500032, India jawahar@iiit.ac.in

[2] Dept. of Computer Science and Engineering, University college of Engineering, Osmania University, Hyderabad-500007, India hafezsyr@ieee.org

II. VISUAL SERVO CONTROL AS A MINIMIZATION PROBLEM

It was shown in [8] that camera pose alignment problem can be presented a minimization problem of a suitable 2D or 3D cost function. We follow here the same context to represent the visual servoing problem.

The positioning task is to move the robot end-effector from an initial pose $P \in R^3 \times SO(3)$ to reach a desired pose P^*. In other words, the problem is to minimize an error vector $e(s)$ of visual features $s(P)$ by finding a vector ΔP that minimizes a cost function $E(s(P))$. Viewing the problem as a nonlinear least squares minimization allows us to formulate the following cost function

$$E(s(P)) = \frac{1}{2}(s(P) - s(P^*))^T(s(P) - s(P^*)). \quad (1)$$

In the remaining of this section, two minimization methods will be reviewed. The first one is based on the first order Taylor series approximation of the cost function. This method is called the gradient decent minimization. The second method is based on the second order Taylor series approximation of the cost function. This is called the Newton minimization.

A. Gradient Decent Minimization

Here, an approximation, using the first order derivative, of the above cost function is evaluated at the desired pose P^*. The cost function given in (1) can be written as

$$E(s(P^*)) \approx E(s(P)) + \frac{\partial E(s(P))}{\partial P}\Delta P. \quad (2)$$

The gradient of the cost function $\frac{\partial E(s(P))}{\partial P}$ is given as

$$\frac{\partial E(s(P))}{\partial P} = (s(P) - s(P^*))^T \ \frac{\partial s(P)}{\partial P} = e(s)^T J(P), \quad (3)$$

where $e(s) = s(P) - s(P^*)$. The criteria in gradient decent minimization is to move in the direction opposite to the gradient. Indeed, the required change in the pose is

$$V = \Delta P = -\lambda J^T(P)e(s), \quad (4)$$

where λ is a positive constant parameter that defines the step size of the minimization process.

This method is known in the robot literature as the Jacobian transpose control method (JTC), and was used in [9], [10] and recently restated in [11]. However, gradient decent methods are known to have a slow and linear convergence rate. The notion of JTC will be used in the remaining of this paper to call all gradient methods.

B. Newton Minimization

Here, an approximation using the second order derivative of the cost function is evaluated at the desired pose P^*. The cost function given in (1) can be written as

$$E(s(P^*)) \approx E(s(P)) + \frac{\partial E(s(P))}{\partial P}\Delta P + \frac{1}{2}\Delta P^T \frac{\partial^2 E(s(P))}{\partial P^2}\Delta P. \quad (5)$$

The first order term $\frac{\partial E(s(P))}{\partial P}$ is as given in (3), while the second order term $\frac{\partial^2 E(s(P))}{\partial P^2}$ is given by

$$\frac{\partial^2 E(s(P))}{\partial P^2} = J^T(P)J(P) + \sum_{k=0}^{n} H_k(P)e_k(s). \quad (6)$$

The matrix H_k is the Hessian matrix of the function $e_k(s)$. In Gauss-Newton minimization, the second order derivative is approximated by

$$\frac{\partial^2 E(s(P))}{\partial P^2} = J^T(P)J(P). \quad (7)$$

Indeed, the required change in the pose is

$$V = \Delta P = -\lambda J^+(P)e(s), \quad (8)$$

where the matrix

$$J^+(P) = \Big((J^T(P)J(P)\Big)^{-1} J^T(P)$$

is the pseudo-inverse of the matrix J. This method is known as the Jacobian Pseudo-inverse control method (JPC), and is widely used in the robot control and visual servoing [12], [11]. In the remaining of this paper, this notion will be used to call all Jacobian Pseudo-inverse methods.

C. Second order Approximation of the Jacobian Matrix

The minimization methods which are presented in sections II-A and II-B are known to have a low convergence rate due to its approximation upto the first order derivative. Malis proposed in [11] an efficient second order minimization method based on the second order approximation of the Jacobian matrix itself. Without going into further details, the second order approximation of the error function can be written as a function of the mean of both, the Jacobian at the desired pose $J(P^*)$ and the current Jacobian $J(P)$.

$$e(s) = -\frac{1}{2}(J(P) + J(P^*))\Delta P, \quad (9)$$

where the approximated Jacobian matrix is given by:

$$\hat{J}(P) = \frac{1}{2}(J(P) + J(P^*)). \quad (10)$$

The required change in the pose given in (4) becomes after introducing the second order approximation

$$\Delta P = -2\lambda\Big(J(P) + J(P^*)\Big)^T e(s). \quad (11)$$

This will be called in the remaining of this paper as Transpose of the Mean of the Jacobian Control method (TMJC).

The required change in the pose given in (8) can be written after introducing the second order approximation as

$$\Delta P = -2\lambda\Big(J(P) + J(P^*)\Big)^+ e(s). \quad (12)$$

This will be called in the remaining of this paper as Pseudo-inverse of the Mean of the Jacobian Control method (PMJC).

This second order approximation of the Jacobian matrix was used for the first time in [13]. Recently, a theoretical justification has been presented in [11].

III. 2D VS. 3D OBJECTIVE FUNCTION

Based on the type of the visual features $s(P^*)$ used in the minimization process, the cost function defined in (1) varies from $E_{2D}(P)$ for 2D visual features from image space to $E_{3D}(P)$ for 3D visual features from the Cartesian space.

If we consider the 2D coordinates of a set of image points as features, the vector $s(P)$ becomes $s_{2D}(P) = [x_1, y_1, \ldots, x_N, y_N]^T$ while the desired vector $s(P^*)$ is $s_{2D}(P^*) = [x_1^*, y_1^*, \ldots, x_N^*, y_N^*]^T$. Indeed, Equation (1) is rewritten as

$$E_{2D}(s(P)) = \frac{1}{2} e_i(P)^T e_i(P), \tag{13}$$

where

$$e_i(P) = s_{2D}(P) - s_{2D}(P^*). \tag{14}$$

In contrast, 3D visual features such as the position and orientation can be part of the feature vector $s_{3D}(P) = [T_x, T_y, T_z, u\theta]^T$. The desired features are $s_{3D}(P^*) = [T^*, u\theta^*]^T$. Similarly to (13) and (14), we can write

$$E_{3D}(s(P)) = \frac{1}{2} e_p(P)^T e_p(P), \tag{15}$$

where

$$e_p(P) = s_{3D}(P) - s_{3D}(P^*). \tag{16}$$

It can be proved that minimizing either $E_{3D}(s(P))$ or $E_{2D}(s(P))$ are equivalent. In other words, $E_{3D}(s(P^*)) = E_{2D}(s(P^*)) = 0$.

Lemma 1: The cost function $E_{3D}(s(P)) = 0$ if and only if the cost function $E_{2D}(s(P)) = 0$.

For the detailed proof reader may referred to [8].

Unfortunately, minimizing the cost function $E_{3D}(s(P))$ has a contradicted behavior with minimizing $E_{2D}(s(P))$. Hybrid methods aim at reducing the undesirability while keeping maximizing the advantages of each of the two function minimization.

A. Model-based Jacobians Computation

In case of the CAD model of the object is available, The set of 3D features can be selected as $s_{3D}(P) = [\ {}^{C^*}T_C,\ {}^{C^*}(u\theta)_C]^T$ while $s_{3D}(P^*) = 0_{(6\times1)}$.

The matrix $J_p(P) = \frac{\partial(e_p(P))}{\partial P}$ is the Cartesian Jacobian matrix that is given by

$$J_p(P) = \begin{bmatrix} R_C^{C*} & 0_{(3\times3)} \\ 0_{(3\times3)} & L_w \end{bmatrix}, \tag{17}$$

where $L_w = \dfrac{\partial(u\theta)}{\partial P}$.

The matrix $J_i(P) = \frac{\partial(e_i(P))}{\partial P}$ is the image Jacobian matrix which is given as follows

$$J_i(P) = \begin{bmatrix} J_{i1}^T & \ldots & J_{i2}^T & \ldots & J_{i3}^T \end{bmatrix}^T,$$

where J_{ik} is computed as a function of the kth image point coordinates x_k, y_k and its depth Z_k as following

$$J_{ik} = \left[\frac{1}{Z_k} \mathbf{S}(x,y) \quad \mathbf{Q}(x,y) \right] =$$

$$\begin{bmatrix} -\frac{1}{Z_k} & 0 & \frac{x_k}{Z_k} & x_k y_k & -(1+x_k^2) & y_k \\ 0 & -\frac{1}{Z_k} & \frac{y_k}{Z_k} & 1+y_k^2 & -x_k y_k & -x_k \end{bmatrix}. \tag{18}$$

The above definition of the Jacobian matrix uses the full 3D information of the object. The depth of points and the relative object-camera pose are required. The control algorithm using the error function and features presented above is called later on in this paper as the model-based JPC. Figure 1(b) illustrates the $s_{3D}(P)$ used in this case.

B. Model-free Jacobians Computation

In case of these 3D information are not available, the vector of 3D features can be set as $s_{3D}(P) = [\ {}^{C^0}T_C,\ {}^{C^0}(u\theta)_C]^T$ *i.e.* the relative pose between the current camera frame F_C and the initial one F_{C^0}. The desired features can be set in this case as $s_{3D}(P^*) = [\ {}^{C^0}T_{C^*},\ {}^{C^0}(u\theta)_{C^*}]^T$*i.e.* the relative pose between the desired current camera frame F_{C^*} and the initial one F_{C^0}. The set of desired features $s_{3D}(P^*)$ can be computed in an off-line step by decomposing the homography or the essential matrix [2], [14].

We use tracking algorithm to estimate the camera motion between the initial and current images. This algorithm, uses epipolar constraints from multiple view geometry as a likelihood function to estimate the *posteriori* distribution of the relative camera pose between the current and initial camera states. Details of this algorithm is beyond the scope of this paper. This relative pose is nothing but the feature vector $s_{3D}(P)$. The *apriori* camera motion distribution is computed using either the velocity control signal commanded by the visual controller to the robot joint controller or the odometery measurements. In other words, the tracking algorithm iterates between predicting the relative camera motion using the velocity control signal and correcting using epipolar geometry constraints.

The image Jacobian $J_i(P)$ can still be computed as given in (18) since the depth, given the relative pose between two cameras, can be estimated as explain in [15]. The Cartesian Jacobian $J_p(P)$ is given in this case by

$$J_p(P) = \begin{bmatrix} {}^{C^0}R_C & 0_{(3\times3)} \\ 0_{(3\times3)} & {}^{C^0}(L_w)_C \end{bmatrix}, \tag{19}$$

while the error function is

$$e_p(P) = s_{3D}(P) - s_{3D}(P^*) = \begin{bmatrix} {}^{C^0}T_C - {}^{C^0}T_{C^*} \\ {}^{C^0}(u\theta)_C - {}^{C^0}(u\theta)_{C^*} \end{bmatrix}. \tag{20}$$

The control algorithm using the error function and features presented above is called later on in this paper as the model-free JPC. Figure 1(a) illustrates the $s_{3D}(P)$ used in this case.

IV. MINIMIZING A HYBRID ERROR FUNCTION

Let us define a hybrid cost function as the weighted sum of the two $E_{3D}(s(P))$ and $E_{2D}(s(P))$ function as follows

$$E_h(s(P)) = \lambda_1\ E_{2D}(s(P)) + \lambda_2\ E_{3D}(s(P)). \tag{21}$$

Here, λ_1 and λ_2 are positive scalar factors. It can be easily shown, using *Lemma 1*, that the hybrid function given in (21)

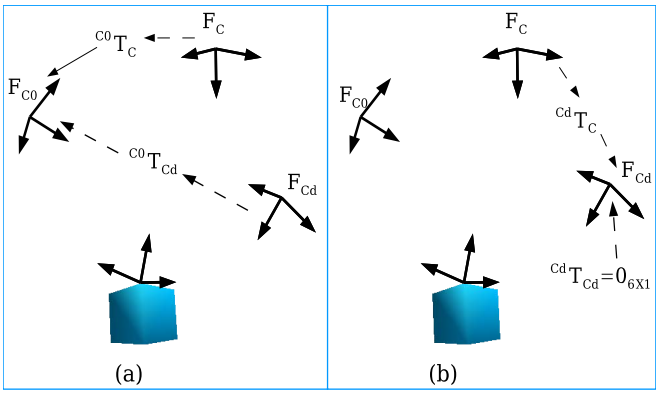

Fig. 1. The current and desired 3D feature vectors. The model-free method, $s_{3D}(P) = [\ {}^{C^0}T_C,\ {}^{C^0}(u\theta)_C]^T$ in (a). The model-based method, $s_{3D}(P) = [\ {}^{C^*}T_C,\ {}^{C^*}(u\theta)_C]^T$ in (b).

is nullified only when $P = P^*$. The two constant λ_1 and λ_2 play the role of the step size of the minimization process in addition to the integration ratio between 2D and 3D spaces. While minimizing this cost function, the process searches for a solutions that reduce the value of the two individual functions $E_{3D}(s(P))$ and $E_{2D}(s(P))$. In the following two subsections we will show how the minimization methods presented in Section II can be used to minimize this hybrid cost function.

A. Gradient Decent Method

Consider a positioning task to be achieved using the gradient decent minimization method. The hybrid cost function given in (1) will be rewritten as

$$E_h(s(P)) = \frac{\lambda_1}{2}\ E_{2D}(s(P)) + \frac{\lambda_2}{2}\ E_{3D}(s(P)), \qquad (22)$$

where the division by 2 is useful to simplify the derivation process. The gradient vector of this cost function is given as

$$\begin{aligned}\frac{\partial E_h(P)}{\partial P} &= \frac{\lambda_1}{2}\frac{\partial\Big(e_i(P)^T e_i(P)\Big)}{\partial P} + \frac{\lambda_2}{2}\frac{\partial\Big(e_p(P)^T e_p(P)\Big)}{\partial P},\\ &= \lambda_1 e_i(P)^T\frac{\partial\Big(e_i(P)\Big)}{\partial P} + \lambda_2 e_p(P)^T\frac{\partial\Big(e_p(P)\Big)}{\partial P},\\ &= \lambda_1 e_i(P)^T J_i(P) + \lambda_2 e_p(P)^T J_p(P). \qquad (23)\end{aligned}$$

To compute the change in the pose that minimizes the above cost function, the change should be in the opposite direction to the gradient. Indeed, the change in the pose is given as

$$\Delta P = -\lambda \mathcal{J}^T(P)\begin{bmatrix} e_i(P) \\ e_p(P)\end{bmatrix}, \qquad (24)$$

where the matrix $\mathcal{J}(P)$ is given

$$\mathcal{J}(P) = \begin{bmatrix} J_i(P) \\ J_p(P)\end{bmatrix}. \qquad (25)$$

In most practical situations we can set $\lambda_1 = \lambda_2 = \lambda$. One may note that local minima may occurs if

$$\begin{bmatrix} e_i(P) \\ e_p(P)\end{bmatrix} \in Ker(\mathcal{J}(P)^T).$$

This means that either $e_i(P) = e_p(P) = 0$, *i.e.* the global minima case, or (i) $e_i(P) \in Ker(J_i^T(P))$ and (ii) $e_p(P) \in Ker(J_p^T(P))$ at the same time. The case (ii) hold for global minima only where $Ker(J_p^T(P)) = \{0_{(6\times1)}\}$. In other words, problem relating to local image minima and Jacobian singularity will not affect the convergence.

B. Gauss-Newton Method

To achieve the minimizing task defined by the hybrid cost function $E_h(P)$ given in (21), we substitute in (8) to get

$$\Delta P = -\lambda \mathcal{J}^+(P)\begin{bmatrix} e_i(P) \\ e_p(P)\end{bmatrix}, \qquad (26)$$

where the matrix $\mathcal{J}(P)$ is given

$$\mathcal{J}(P) = \begin{bmatrix} \frac{\partial e_i(P)}{\partial P} \\ \frac{\partial e_p(P)}{\partial P}\end{bmatrix} = \begin{bmatrix} J_i(P) \\ J_p(P)\end{bmatrix}. \qquad (27)$$

Here,

$$\mathcal{J}^+(P) = \Big(J_i^T J_i + J_p^T J_p\Big)^{-1}\Big[J_i^T J_p^T\Big]. \qquad (28)$$

Since $Ker(J_p^T(P)) = \{0_{(6\times1)}\}$, the velocity vector will be nullified only at the desired position without suffering from any local minima.

To summarize, It is shown that minimizing the hybrid proposed function is realizable. This function $E_h(P)$ has a common global minima with each of $E_{2D}(P)$ and $E_{3D}(P)$. The minimization of $E_h(P)$ is done using NLSM owing to the non-linear part arising from the 2D image space. However, global minima is the optimal solution that can be asymptotically obtained after a limited number of iterations.

C. Computing the Factors λ_1 and λ_2

The integration between 2D and 3D can be biased using the scalar factors λ_1 and λ_2. The control law can be written when $\lambda_1 \neq \lambda_2$ as

$$V = \Delta P = -\lambda \mathcal{J}(P)^+\begin{bmatrix} \Gamma_1 & \Gamma_2\end{bmatrix}\begin{bmatrix} e_i(P) \\ e_p(P)\end{bmatrix}, \qquad (29)$$

where $\Gamma_1 = diag(\lambda_1)$ and $\Gamma_2 = diag(\lambda_2)$. This gives a sense in designing a high level rule that computes these factors λ_1 and λ_2. A proper computation of the integration factors can be used to improve the performance of the whole process by individually taking advantages of the 2D or the 3D features used in the error function. For example, it is possible to put $\lambda_1 = \frac{\omega_{2D}}{\omega_{2D}+\omega_{3D}}$ and $\lambda_2 = \frac{\omega_{3D}}{\omega_{2D}+\omega_{3D}}$. The weights λ_1 and λ_2 are the importance factors of the 2D and 3D informations respectively [16]. Another method to adjust the weights λ_1 and λ_2 is the function proposed in [17]. In this method, the process starts with PBVS *i.e.* corresponds to higher value of λ_2, and end with IBVS *i.e.* corresponds to higher value of λ_1. Ending with image-based avoids the effect of noisy data on the stability of 3D algorithm.

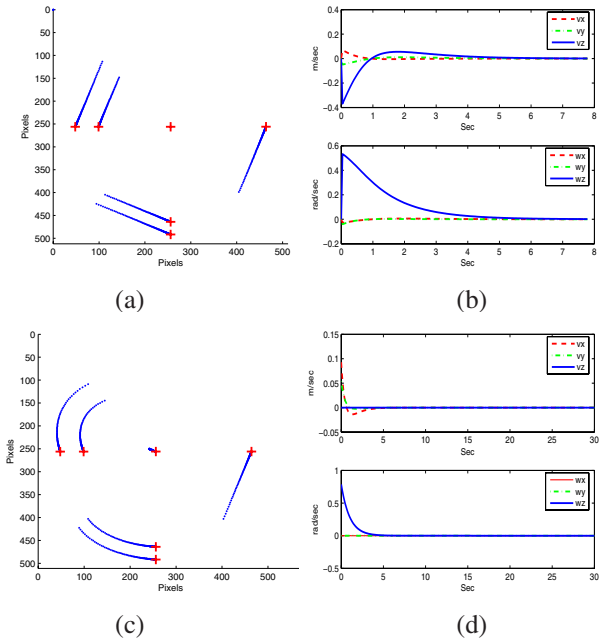

Fig. 2. The image features trajectories and the screw velocity. Image-based visual servoing in (a) and (b). $21/2$D visual servoing (c) and (d). The desired positions of the image features are marked by +.

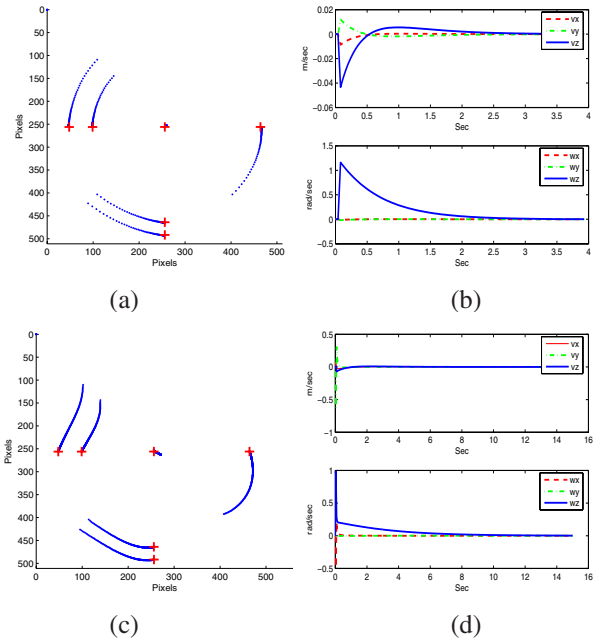

Fig. 3. The image features trajectories and the screw velocity of JPC visual servoing. In (a) and (b), the model-based method. In (c) and (d), the model-free method. The desired positions of the image features are marked by +.

V. EXPERIMENTAL EVALUATION

We present simulation experiments where the proposed methods are compared to previous method like IBVS, PBVS, and hybrid methods, namely $21/2$D visual servoing. These methods in addition to our proposed methods are implemented in a simulation framework. The simulation assumes a perspective camera model with $1000m$ focal length and unit aspect ratio.

The comparison is carried out for two positioning tasks. First one is with a $\pi/4$ rad pure rotation error around the camera optical axis. Using this task we will evaluate the efficient performance of the proposed methods compared to previous work in the literature like IBVS and its improved $21/2$D VS. The task is achieved using our two (model-based and model-free) JPC proposed methods, and the results are compared to IBVS and $21/2$D VS. The camera retreat using each method is evaluated by the magnitude of the screw velocity V_z along the camera optical axis.

The second task is a general positioning task that contains a rotational and translational errors. The task is useful to evaluate the camera trajectory in the Cartesian space. The camera trajectories using our two JPC methods and methods like PBVS and $21/2$D VS are compared here. The servoing target object consists of six non-planar points.

A. Camera Retreat from Rotation Error

The experiments for the first task is done using JPC (model-based and model-free), IBVS and $21/2$D. The results are shown in Figs. 2, and 3.

In case of using IBVS, the camera retreat along the camera optical axis is demonstrated in Figs. 2(a) and 2(b). These figures show the image trajectory and screw velocity respectively. A nice image trajectory was obtained along with a considerable pure backward translation motion (camera retreat). The task is done properly using $21/2$D. This is shown in Figs. 2(c) and 2(d). It is clear that there is no significant translational motion while the rotation error decreased properly to zero. In other words, the task has been done without camera retreat.

The results in case of using our proposed (model-based and model-free) JPC methods are shown in Figs. 3(a), 3(b) and Figs. 3(c), 3(d) respectively. The model-based JPC method shows an image trajectory similar to the one obtained using $21/2$D method. However, a moderately small camera retreat exists. A similar image trajectory is obtained using the model-free JPC but with very small camera retreat which can be easily ignored.

B. Camera trajectory in the Cartesian Space

Now, we consider the second task. This is a general task with both translational and rotational motion. First we present the results from PBVS where the camera trajectory in the Cartesian space is a straight line. This is the shortest camera path. Figs 4(a) and 4(b) show the feature trajectory in the image space and camera trajectory in the Cartesian space respectively. However the image trajectory is an undesirable complex curve and some features got out of the camera field of view. In contrast, $21/2$D visual servoing, as shown in Figs 4(c) and 4(d), has improved the image trajectory, but nothing about the Cartesian camera path, it is not straight line at all. Using model-based PJC method produces a camera trajectory that is not only very much near to the straight line, but also with fine image trajectory. the model-free JPC method produces a similar image trajectory, and camera path is approximately straight line.

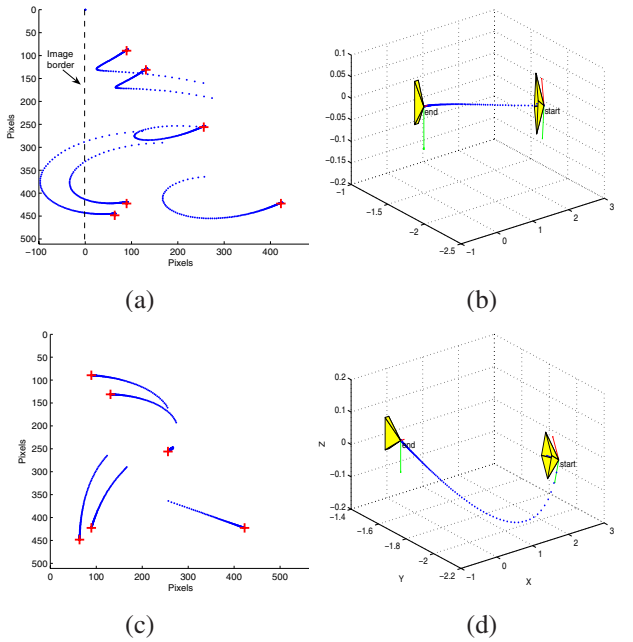

Fig. 4. The image features trajectories and the camera trajectory in the Cartesian space. PBVS visual servoing in (a) and (b), 21/2D visual servoing in (c) and (d). The desired features are marked by +.

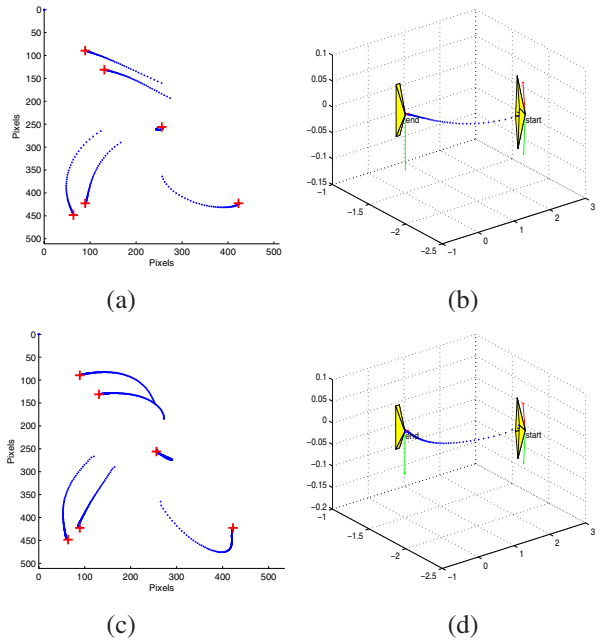

Fig. 5. The image features trajectories and the camera trajectory in the Cartesian space of JPC visual servoing. In (a) and (b), the model-based method. In (c) and (d), the model-free method. The desired features are marked by +.

In general, our both model-based and model-free methods show an improved performance in image space and Cartesian space together. In both methods, image features are less probable to leave the image while the camera performs less retreat in the Cartesian space. An improvement in the features behavior in the image space for all considered points exists. Most of previous hybrid methods improve the path of one point in the image.

VI. CONCLUSION

A novel 5D visual servoing method is formulated as a minimization process. Applying gradient decent minimization gives the 5-D Jacobian Transpose Control method (JTC). From Gauss-Newton minimization we obtain the 5-D Jacobian Pseudo-inverse Control method (JPC). The second order approximation of the Jacobian matrix is also considered here. The results show that our proposed methods improve the performance of all features in the image space as while as the camera trajectory in the Cartesian space. Future work will focus on the visibility and joint limits constraint. They will be introduced to the minimization process. The visual servoing process will be considered as a constrained minimization process.

REFERENCES

[1] P. Corck and S. Hutchinson, "A new partitioned approach to image-based visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 4, pp. 507–515, Aug 2001.

[2] E. Malis, F. Chaumette, and S. Boudet, "2 1/2 d visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 238–250, April 1999.

[3] S. Hutchinson, G. Hager, and Cork, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 18–27, 1996.

[4] F. Chaumette and S. Hutchinson, "Visual servo control, part i: Basic approaches," *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, December 2006.

[5] ——, "Visual servo control, part ii: Advanced approaches," *IEEE Robotics and Automation Magazine*, vol. 14, no. 1, March 2007.

[6] E. Malis and P. Rives, "Robustness of image-based visual servoing with respect to depth distribution errors," in *IEEE Int. Conf. on Robotics and Automation, ICRA'03*, vol. 1, Taipei, Taiwan, Sept. 2003, pp. 1056–1061.

[7] E. Malis and F. Chaumette, "Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods," *IEEE Trans. on Robotics and Automation*, vol. 18, no. 2, pp. 176–186, April 2002.

[8] A. H. Abdul Hafez and C. V. Jawahar, "Improvement to the minimization of hybrid error functions for pose alignment," in *IEEE Int. Conf. on Automation, Robotics, Control, and Vision, ICARCV'06*, Singapore, December 2006.

[9] K. Hashimoto and H. Kimura, "L q optimal and nonlinear approaches to visual servoing," in *Visual Servoing*, ser. World Scientific Series in Robotics and Automation Systems. World Scientific Press, 1993, vol. 7, pp. 165–198.

[10] E. Malis, "Hybrid vision-based robot control robust to large calibration errors on both intrinsic and extrinsic camera parameters," in *European Control Conference, ECC'01*, Porto, Portugal, September 2001, pp. 2898–2903.

[11] ——, "Improving vision-based control using efficient second-order minimization techniques," in *IEEE Int. Conf. on Robotics and Automation, ICRA'04*, New Orleans, USA, April 2004.

[12] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Trans. on Robotics and Automation*, vol. 8, no. 3, pp. 313–326, July 1992.

[13] O. Tahri and F. Chaumette, "Application of moment invariants to visual servoing," in *IEEE Int. Conf. on Robotics and Automation, ICRA'03*, vol. 3, Taipeh, Taiwan, May 2003, pp. 4276–4281.

[14] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2003.

[15] A. H. Abdul Hafez and C. V. Jawahar, "Depth/model estimation using particle filters for visual servoing," in *IEEE Int. Conf. on Pattern Recognition, ICPR'06*, Hong Kong, August 2006.

[16] ——, "Probabilistic integration framework for improved visual servoing in image and cartesian spaces," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'06*, Bejing, China, October 2006.

[17] ——, "Probabilistic integration of 2d and 3d cues for visual servoing," in *IEEE Int. Conf. on Automation, Robotics, Control, and Vision, ICARCV'06*, Singapore, December 2006.