COMPARING VISUAL SERVOING ARCHITECTURES FOR A PLANAR ROBOT

P. Gonçalves*, P. Ferreira†, P. Pinto†, A. Oliveira†, G. Brito†, L. Pina†, J. Pinto†

*Escola Superior de Tecnologia de Castelo Branco,
Av. Empresário, 6000-767 Castelo Branco, Portugal
†IDMEC/IST - Technical University of Lisbon - Instituto Superior Técnico,
Av. Rovisco Pais, 1049-001 Lisboa, Portugal
e-mail: paulo.goncalves@est.ipcb.pt

Keywords: 2D Visual Servoing; 2½D Visual Servoing; Path Planning; Machine Vision; Robotic manipulators.

Abstract

This paper presents a study of three different Visual Servoing architectures for a planar robot manipulator, comparing their behaviour in achieving a desired goal. The studied architectures were 2D Visual Servoing, 2½D Visual Servoing and Visual Servoing by path planning. The target used in the experiments was planar. Four target points were selected as the visual features. Since the robot is planar, has no wrist and the visual features are coplanar, some simplifications could be made in the 2½D Visual Servoing architecture. The other architectures studied were implemented with no simplifications, with the exception of the joint limits potential of the path planning visual servoing. The study presented is based on a 2 d.o.f. planar robot manipulator constructed at Instituto Superior Técnico [1,9], Mechanical Engineering Department, Robotics and Automation Laboratory.

Planar robots are widely spread in the industry, two of their most important characteristics being speed and precision. This type of robotic structure can easily evolve to a SCARA robot, used in vertical assembly tasks, palletizing, etc.

The objective of this work is to study the referred robotic manipulator under 2D Visual Servoing, 2½D Visual Servoing and Visual Servoing by path planning. The robot, in an eye-in-hand configuration, should move from an initial to a final position, with the control taking place in the image features space. The control laws and the control structures used to accomplish these objectives are presented in section 2. The simulation results are presented in section 3. Some of the conclusions of this work are briefly exposed in section 4.

2 Theoretical Framework

Machine Vision and Robotics can be used together to control a robot manipulator. This type of control, defined as Visual Servoing, uses visual information from the work environment to control a robot manipulator performing a task.

The visual information can be obtained by two ways: using direct information from the image (2D visual servoing); or using 3D information of the object from the image(s) (3D visual servoing). The second case needs an on-line processing for pose calculation. A good explanation of the differences is done in [7].

1 Introduction

In this paper we control, based on visual servoing, a two degrees of freedom planar robotic manipulator. The robot in use was constructed at Instituto Superior Técnico [1,9], Mechanical Engineering Department, Robotics and Automation Laboratory.
The needed visual features to move the robot manipulator are really just different representations of the same point. It is necessary to define: a 3D point, \( P \); an image point in metric coordinates, \( m \); a modified image point in metric coordinates, \( m_e \); an image point in pixel coordinates, \( p \); an image features vector, \( s \); a desired image features vector, \( s^d \); and a error vector, \( e \).

\[
P = [X \ Y \ Z \ 1]^T \quad (1)
\]
\[
m = \frac{X}{Z} \ Y \ Z \ 1]^T \quad (2)
\]
\[
m_e = \frac{X}{Z} \ Y \ Z \ 1]^T \quad (3)
\]
\[
p = A \cdot m = [u \ v \ 1]^T \quad (4)
\]
\[
s = [p_1 \ ... \ p_k \ ... \ p_N]^T \quad (5)
\]
\[
s^d = [p_1^d \ ... \ p_k^d \ ... \ p_N^d]^T \quad (6)
\]
\[
e = s - s^d \quad (7)
\]

where, \( N \) is the number of visual features, and \( A \) the intrinsic parameters matrix of the camera:

\[
A = \begin{bmatrix} 
   f \cdot k_u & f \cdot k_v \cdot \cot(\theta) & u_o \\
   0 & f \cdot k_v \cdot \sin(\theta) & v_o \\
   0 & 0 & 1 
\end{bmatrix} \quad (8)
\]

2.1 Visual Servoing 2D

Following the approach to Visual Servoing made in [4], to obtain a “good control” it is necessary to minimize a function \( f(s) \):

\[
f(s) = \frac{1}{2} \| s - s^d \|^2 \quad (9)
\]

When the desired image features are obtained, we have \( e = 0 \).

Considering \( r \) the pose of the end-effector (translation and rotation), dependent on the robot joint variables \( q \), it is then necessary to find a robot pose that minimize \( f(s) \):

\[
\min_{q} f(q) \quad (10)
\]

The function \( f(s) \), reaches a minimum when its first derivate goes to zero:

\[
\frac{\partial f}{\partial s} \cdot \frac{\partial s}{\partial r} \cdot \frac{\partial r}{\partial q} = 0 \quad (11)
\]

\[
\frac{\partial s}{\partial r} \equiv J_i, \text{ is the image Jacobian.}
\]

\[
\frac{\partial r}{\partial q} = J_r(q), \text{ is the robot Jacobian.}
\]

\[
\frac{\partial r}{\partial q} = J_r(q) \quad (12)
\]

\[
\dot{s} = J_i \cdot \dot{r} \quad (13)
\]

According to [5], if the two Jacobians are not singular the solution to the minimization problem becomes:

\[
\frac{\partial f}{\partial s} = 0, \text{ i.e., } e = 0
\]

and if the object velocity, at the desired position, is zero, i.e., \( \dot{s}_d = 0 \):

\[
\frac{\partial e}{\partial t} = \frac{\partial s}{\partial t} = \frac{\partial s}{\partial r} \frac{\partial r}{\partial t} \quad (14)
\]

where:

\[
\frac{\partial r}{\partial t} \equiv \dot{r}, \text{ the end-effector velocity.}
\]

As seen in equation (14), to obtain a control law it is necessary to obtain a relation between the image features and the image features velocities. By definition of forward finite differences, [12], the image features velocity becomes:

\[
\frac{\partial s}{\partial t} = \frac{1}{h} \cdot (s_{t+h} - s_t) + \mathcal{O}(h^2) \quad (15)
\]

the above expression can be approximated, in order to obtain our goal:

\[
\dot{s} = \frac{1}{h} \cdot (s^d - s) \quad (16)
\]

where \( h \) is a constant that results from the sample time. It can be seen that equation (16), produces some errors, especially if the desired features, \( s^d \),
are far away from the actual position, \( s \). Moreover, this equation is only valid locally.

From equations (14) and (16), the following relations can be obtained:

\[
\frac{\partial e}{\partial t} = J_i \cdot \dot{r} \Rightarrow s - s^d = -h \cdot J_i \cdot \dot{r}
\]

\[
\dot{r} = -\frac{1}{h} \cdot J_i^\top \cdot (s - s^d)
\]  
(17)

The image Jacobian is calculated using the Pin-Hole camera model [2]:

\[
J_i = A \cdot \begin{bmatrix}
-\frac{1}{Z} & 0 & \frac{X}{Z^2} & x \cdot y & -(1 + x^2) & y \\
0 & -\frac{1}{Z} & \frac{Y}{Z^2} & 1 + y^2 & -x \cdot y & x \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]  
(18)

The robot Jacobian is calculated using the methods in [13].

The end-effector pose \( r \), can be obtained directly:

\[
r = -\int \frac{1}{h} \cdot J_i^\top (r) \cdot e \, dt
\]

The exponential decayment of the error function (9) can be verified, using equations (14) and (16):

\[
\dot{e} = -K \cdot e
\]  
(19)

where: \( K = \frac{1}{h} \)

The condition expressed in equation (19), that other authors stated as a condition \([4,8,11]\), here is stated as a result when deriving the control law. This method makes very clear the verification that this condition is only valid in a neighborhood, function of the sample time \( h \), of the actual image feature position, \( s \).

Then, the visual control law is:

\[
r = \int J_i^\top (r) \cdot \dot{e} \, dt
\]  
(20)

### 2.2 Visual Servoing 2½ D

In the work done by Malis [8] the 2D Visual Servoing has evolved to 2½D Visual Servoing, in which 3D information from the target is used for control. The main goal achieved was the global stability of servoing. Comparing with 2D visual servoing the increase of information used was the depth relation of the target and the rotation, between the actual, \( a \), and desired, \( d \), camera poses.

The error vector then becomes:

\[
e = \begin{bmatrix}
u^a - u^d \\
v^a - v^d \\
\log \left( \frac{Z^a}{Z^d} \right)
\end{bmatrix}
\]  
(21)

In 2½D visual servoing architecture, the depth relation and the rotation are computed using homographies or the fundamental matrix [8]. In our
particular case of study, a planar robot, the depth relation can be obtained in a simple way, taking into account the pin-hole camera model used and the fact that the camera always moves in the same plane, see Fig. 3.

\[
\frac{Z^u}{Z^d} = \left| \frac{V^d}{V^u} \right| \tag{24}
\]

2.3 Visual Servoing by Path Planning

The path planning architecture studied follows the work in [10]. An image features trajectory is computed off-line, in order to be introduced on the visual control loop as the reference to be followed.

This method for obtaining the image features trajectory is divided in three main parts. The first one is the data initialization, in which the initial pose is obtained from the initial and desired robot position. The second part consists of an iteration process in which the image features discrete path is composed. The third part consists in obtaining the final trajectory of the image features over time.

During initialization, the image points corresponding to the image features as seen in the initial and desired robot position are obtained. The translational vector and rotational matrix between the object and the initial/desired position are computed [10] using a pose estimation method [3]. The initialization ends with the computation of the initial pose relative to the desired one, from the resulting data. In this paper, the initial pose was computed analytically, assuming that the object referential is known. That is possible in simulation, and has the advantages of greatly reducing the simulation time as well as of eliminating the errors due to the pose estimation algorithm.

\[
\begin{align*}
{d}_R &= {d}_R_0 \cdot {i}_R_0^T \\
{d}_t &= -{d}_R_0 \cdot {d}t + {d}t_0 \\
\gamma^T &= \left[ {d}t \left( u \theta \right)^T \right]
\end{align*}
\tag{25}
\]

The iteration process in which the image features discrete path is composed begins with the estimation of the next pose, \( \gamma_{k+1} \), from the actual pose, \( \gamma_k \), and a composed force, \( F \). The composed force is a sum of two different forces, an attractive, \( F_a \), and a repulsive, \( F_r \).

\[
\begin{align*}
\gamma_{k+1} &= \gamma_k + \varepsilon_h \cdot \frac{F(\gamma_k)}{||F(\gamma_k)||}, \varepsilon_h > 0 \\
F(\gamma_k) &= F_a(\gamma_k) + F_r(\gamma_k)
\end{align*}
\tag{26}
\]

The attractive force minimizes the trajectory and the repulsive force is set to avoid, only, the physical image limits.

\[
\begin{align*}
F_a &= \frac{1}{2} \cdot a \cdot \left\| \gamma - \gamma_a \right\|^2, a > 0 \\
F_r(\gamma) &= -\nabla(\gamma_a) = -a \cdot \gamma, a > 0
\end{align*}
\tag{27}
\]

The image repulsive force keeps the image features visible during the path. It is supposed to grow as the image features pass a centered safety square and tend to the image limits, “pushing” them to the center of the image.

\[
V_a = \begin{cases} 
v^2(\gamma) & \text{if } s \text{ is out of the safety zone} \\
0 & \text{if } s \text{ is within of the safety zone}
\end{cases}
\]

where:

\[
v(\gamma) = \prod_{j=1}^n \left( \left( u(j) - u_{a min}^n \right) \left( u(j) - u_{a min}^n \right) \left( v(j) - v_{a min}^n \right) \left( v(j) - v_{a min}^n \right) \right)
\]

\[
u(s) = \prod_{j=1}^n \left( 1 - \frac{u(j)}{u_{max}} \right) \left( 1 - \frac{u(j)}{u_{min}} \right) \left( 1 - \frac{v(j)}{v_{max}} \right) \left( 1 - \frac{v(j)}{v_{min}} \right)
\]

\( n \), number of image features

\[
F_i = \begin{cases} 
-M_j^{-1} \cdot J_i^{-1} \cdot \nabla(\gamma^T) & \text{if } s \text{ is out of the safety zone} \\
0 & \text{if } s \text{ is within of the safety zone}
\end{cases}
\]

and,

\[
M_i = \frac{\partial r}{\partial \gamma} = \begin{bmatrix} 
{d}_R & 0_{3x3} \\
0_{3x3} & L_w^{-1}
\end{bmatrix}
\]

\[
L_w^{-1} = I_{3x3} + \frac{\theta}{2} \cdot \sin^2 \left( \frac{\theta}{2} \right) \cdot S(u) + \left( 1 - \sin(\theta) \right) \cdot \left( S(u) \right)^2
\]
Fig. 1: Image repulsive potential

All the remaining poses until the robot reaches the desired position are estimated in a recursive process, which begins with the initial pose. The attractive force is highest in the initial pose and, if the algorithm is to work properly, it will diminish during the trajectory, tending to zero at the final pose. The recursion ends when the attractive potential reaches a certain value close to zero, meaning that the actual pose is close enough to the desired position. The result of the iteration process is a vector of image features which contains the image variation of each image feature as the camera goes from the initial to the desired position.

The final trajectory of the image features over time must be continuous and differentiable, so was chosen a $C^2$ curve. It is computed from the vector obtained during the iteration process $s^p$, a pre-defined sample time - video rate $\Delta T = t_k - t_{k-1}$, and the final time for the servoing, using a cubic spline.

$$s^p(t) = A_k \cdot t^3 + B_k \cdot t^2 + C_k \cdot t + D_k$$ (28)

where: $(k-1) \cdot \Delta T \leq t \leq (k) \cdot \Delta T$

The control law used follows [10]:

$$\dot{r} = -K_1 \cdot J_i^{-1} \cdot (s^a_k - s^p_k) + K_2 \cdot J_i^{-1} \cdot (s^p_k - s^p_{k-1})$$ (29)

where: $K_2 = \frac{\lambda}{\Delta T}$, and $\lambda$ is a gain.

3 Results

A planar robotic manipulator of two degrees of freedom, constructed at the Mechanical Engineering Department in Instituto Superior Técnico [1,9], was used.

Fig. 2: Planar Robotic Manipulator

The Camera used was a commercial one, with the following matrix $A$ of intrinsic parameters:

$$A = \begin{bmatrix} 716.1905 & 0 & 0 \\ 0 & 742.9787 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

The inner robot controller implements a Cartesian PID control law [13], with the end-effector pose $r$, as input.

A Matlab™ 6.0 simulink model was developed and validated in [6], to implement Visual Servoing architectures. A Zero Order Hold block was added to the block diagram to simulate the time for image acquisition and processing. For all simulations the video rate is 20 (ms), and the Robot inner loop sample time is 1 (ms).

Fig. 3: Planar Robot with attached Camera.
In the following sections one can see the results obtained with the three architectures studied in this paper. The target used has as visual features four coplanar points. All the simulations were based in the following initial and desired end-effector positions of the robot:

\[ q_i = \begin{bmatrix} -90^\circ & 90^\circ \end{bmatrix} \]

\[ q_d = \begin{bmatrix} -107^\circ & 103^\circ \end{bmatrix} \]

### 3.1 Visual Servoing 2D

The 2D visual servoing architecture exposed in section 2.1, was implemented in Matlab Simulink, using the depth Z at the desired position, for image Jacobian calculus. In the simulation, the gain K=5.

![Fig. 4: 2D Visual Servoing control architecture](image)

![Fig. 5: Image features path in the retina plane for 2D Visual Servoing](image)

![Fig. 6: Image features error between \(s^d\) and \(s^a\) during the 2D Visual Servoing](image)

![Fig. 7: Path of the end-effector during the 2D Visual Servoing](image)

### 3.2 Visual Servoing 2½ D

In the simulation, the gain K=1.
The results for the attractive and repulsive forces are presented in Fig. 12 and Fig. 13. The repulsive force is zero during the process, as the image features are always within the safety zone. The attractive force approaches zero as expected, since the stopping condition was $1e^{-5}$.

In the simulation, we chose the gains $K_1=1/0.02$ and $K_2=5$. The final time for servoing was chosen to be 1.5 (sec). The simulation time has continued for one second more, until the system completely stabilized.

### 3.3 Visual Servoing by Path Planning

The off-line image features path generation was accomplished with 14 iterations of the algorithm.
4 Conclusions and Future Work

In this paper we studied the control of a two degrees of freedom planar robot manipulator by visual servoing, based on a validated experimental apparatus. 2D visual servoing was implemented, and a simple modification when deriving the control equations was used to clarify the locally condition.

We also implemented path planning by visual servoing and a modified 2½D visual servoing architectures.

Both 2D and 2½D Visual Servoing architectures stabilize at 1.5 (sec), but the first one has some undesired oscillatory behavior. For visual servoing by path planning the trajectory time was forced to be 1.5 (sec), in order to be compared with the other two architectures (this is not the best result achieved). Increasing the final time for servoing, 3.5 (sec) for example, the visual servoing by path planning achieves better results, i.e., better precision and smooth curves (reducing the final oscillations and the errors in Fig. 16 and Fig. 17, respectively). Increasing the gain for 2D and 2½D visual servoing the response is quicker but more oscillatory, as expected.

In the path planning architecture, if the final time for the servoing is decreased, the system is not able to track the pre-defined trajectory even if the gain is increased, an action that also might cause oscillations and instability.

With the modified 2½D visual servoing architecture the results improve, when compared with the traditional 2D visual servoing, as seen in the corresponding figures. Using visual servoing by path planning, with the correct final time for the trajectory, the obtained results are comparable with the 2½D architecture. With path planning we are always sure that the image features always remain in the retina plane, due to the repulsive image potential, a result not accomplished by the two other architectures. The worst precision was obtained with 2D visual servoing.

To completely compare the three visual servoing architectures exposed, it is necessary in the future to apply these algorithms to our real planar robot.
Acknowledgements

This work is supported by the “Programa de Financiamento Plurianual de Unidades de I&D (POCTI), do Quadro Comunitário de Apoio III” and by the FCT project POCTI/EME/39946/2001.

References


