Object Servoing of Differential-Drive Robots

Weibin Jia, Wenjie Zhao, Zhihuan Song and Zhengguo Li

1. School of Aeronautics and Astronautics, Zhejiang University, Zhejiang 310027, China
   E-mail: j@zju.edu.cn, zhaowenjie8@zju.edu.cn

2. School of Control Science and Engineering, Zhejiang University, Zhejiang 310027, China
   E-mail: songzhihuan@zju.edu.cn

3. SRO Department of Institute for Infocomm Research, 1 Fusionopolis Way, 138632, Singapore
   E-mail: ezgl@gmail.com

Abstract: Due to possibly changing pose of a movable object and nonholonomic constraint of a differential-drive robot, it is challenging to design an object servoing scheme for the differential-drive robot to asymptotically park at a predefined relative pose to the movable object. In this paper, a novel object servoing scheme is designed for the differential-drive robots. Each on-line relative pose is first estimated by using feature points of the moveable object and it serves as the input of an object servoing friendly parking controller. The linear velocity and angular velocity are then determined by the parking controller. Experimental results validate the performance of the proposed object servoing scheme. Due to its low on-line computational cost, the proposed scheme can be applied for last mile delivery of differential-drive robots to movable objects.

Key Words: object servoing, differential-drive robots, motion estimation, asymptotic stabilization

1 INTRODUCTION

Visual servoing is widely used to drive a differential-drive robot from an initial pose to a goal pose using visual feedback [1, 2]. The goal pose is fixed and it is predefined by a previously acquired image with a pinhole camera. The differential-drive robot is subject to nonholonomic constraint, which makes the visual servoing task challenging, especially when the final pose is required to be accurate. This is because a nonholonomic system cannot be asymptotically stabilized by any time-invariant continuous state feedback control law due to the well-known Brockett necessary condition [3].

Various control methods were developed to asymptotically stabilize the nonholonomic systems such as [4, 5, 6] in the past decades. The singularity of the nonholonomic systems is deemed undesirable by these methods due to loss of controllability. The singularity set of the differential-drive robot is the set of robot poses with the orientation being the same as the goal orientation [4, 5, 6]. The robot is forced to escape from the singularity set by the controllers in [4, 5, 6]. Recently, it was found in [7] that the singularity set of the nonholonomic robot includes a subset which is controllable. The subset is determined by the goal pose and it is named as a “singularity line”. On top of switched control [9, 10, 11], a motion controller was designed in [7] for asymptotic stabilization of the differential-drive robot by taking advantage of singular surfaces of the configuration space through the controllable “singularity line”. Hence, the controller in [7] has a unique feature: it generates straight and smooth motions (which are singular) naturally when the pose of the robot is in the “singularity line”. The robot also always tries to approach the “singularity line” under the control of the controller in [7].

All these controllers assumed that the pose of the robot is already available. This is not true for a visual servoing scheme in which the pose of the robot needs to be estimated. There are two typical ways to address the visual servoing of mobile robots. Position-based visual servoing (PBVS) of mobile robots was investigated in the literature to reduce the visual servoing task to a control problem in the Cartesian space [8, 12]. These approaches require 3-D metric information of the features known as a priori, which is further used to reconstruct full-system state for feedback control. An alternative way is to adopt an image-based visual servoing (IBVS) strategy. Elements of the estimated homography were formulated in [13] as the output of the system, based on which a control law was designed by the input–output linearization technique. A novel motion estimation methodology was proposed in [14] by using correspondences of three unknown feature points in two images to directly compute the relative pose from images, and a 2-1/2-D visual servoing strategy was then proposed to regulate a nonholonomic mobile robot with an onboard camera to its goal pose. It should be pointed out that the goal pose is fixed in the existing visual servoing systems. However, there are many cases that the goal pose is not fixed but being defined as a relative pose to a movable object. For example, a mobile robot is asked to deliver food to a patient in a movable bed and the bed could be pushed to another pose. Since the goal pose is determined by a given object rather than a previously captured image in [1, 2], the new problem is called object servoing. Since the pose of the object can be changed, the existing visual servoing schemes...
are not applicable to the object servoing. It is thus desired to
design an object servoing scheme such that the robot can part
itself with a relative pose to a given object regardless of its
pose.

A novel object servoing scheme is designed on top of an ob-
ject servoing friendly controller in this paper. The controller
is an improved one of the controller in [7] and it is designed
by integrating a fractional-order controller [15] into the con-
troller in [7]. One uniqueness of the fractional-order con-
troller [15] is that the system can be asymptotically stabilized
within a finite time interval. The proposed scheme also in-
cludes a very simple but efficient way to compute the relative
pose for the controller. The goal pose is defined by the given
object in an image which is called a reference image. A deep
learning based method is adopted to detect the given object in
the reference image [16]. Feature points in the given object
are identified and their 3D coordinates are also stored. All
these operations are conducted off-line. It is assumed that
part of the feature points are visible by the onboard camera
of the robot when the robot starts from its initial pose. Each
image captured on line is called a query image. The fea-
ture points of the given object in the reference image are then
matched to those feature points in the query image. The rel-
ative pose of the current pose with respect to the goal one is
then estimated for the robot by using a visual based motion
estimation algorithm. Besides the proposed pose estimation
method, existing pose estimation methods such as [17, 18]
can also be applied to estimate the pose. The complexity of
the algorithm in [17] is an issue for a real time controller
while the accuracy of the algorithm in [18] needs to be im-
proved.

The proposed scheme also includes an object servoing
friendly controller. The relative pose is serve as the input
to the improved controller. The corresponding linear veloc-
ity and angular velocity are then computed by using the im-
proved controller. The robot moves according to the com-
puted linear velocity and angular velocity. A possible issue
for the object servoing is that there might be no common
features between the object in some query images and the
object in the reference image when the robot moves under
the control of the parking controller. This would happen if
the parking controller is not object servoing friendly. Fortu-
nately, the improved controller is much more object servoing
friendly than the controllers in [4, 5, 6]. This is because that
the given object in the reference image is captured by the
onboard camera when the robot is at the goal pose, i.e., the
onboard camera is oriented along the “singularity line”. The
robot always tries to approach the “singularity line” under
the improved controller. Both simulation and experimental
results are used to verify the efficiency of the proposed object
servoing scheme. Besides the proposed scheme, an alterna-
tive scheme is to first estimate the pose of movable object
on-line [23] and then compute the linear velocity and angular
velocity using the improved controller. Compared with such
an alternative one, the on-line computational cost of the pro-
posed scheme is much lower. As such, the proposed scheme
has a good potential to be adopted for the last mile delivery
of differential-drive robots to movable objects.

The rest of this paper is organized as below. A new problem
formulation on object servoing is provided in Section 2. An
object servoing scheme is then proposed in Section 3. Experi-
mental results are given in Section 4 to verify the efficiency
of the proposed scheme. Finally, conclusion remarks are given
in Section 5.

2 Problem Formulation on Object Servoing

A differential-drive mobile robot that is equipped with an
onboard camera is considered in this paper and the robot is
shown in Fig. 1. The motion of the robot is described in
terms of a rotation matrix \( R \in \mathbb{R}^{3 \times 3} \) and a translation vector
\( T \in \mathbb{R}^{3} \) as

\[
R = \begin{bmatrix}
\cos \theta & -\sin \theta & 0 \\
\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{bmatrix}, \quad T = \begin{bmatrix}
t_x \\
t_y \\
0
\end{bmatrix},
\]

where \( \theta \in \mathbb{R} \) is the rotational angle around the z-axis.

Let \( F \) be an orthogonal coordinate system which is attached
to the on-board camera, with its origin being located at the
camera center. The z-axis is supposed to pass through the
midpoint of the wheels axis, and it is orthogonal to the mo-
tion plane of the mobile robot. The z-axis is along the optical
axis, and it is aligned with the front of the robot. In addition,
one more orthogonal coordinate system \( F^* \) is adopted to rep-
resent the goal pose of the camera. For simplicity, \( F \) and \( F^* \)
are called a query image and a reference image, respectively.
A deep learning based method was used off-line to detect the
given object in the reference image [16]. The 3D coordinates
of the feature points in the reference image are stored. All
these operations are conducted off-line. The goal pose is not
fixed but a relative pose to the given movable object such as
a bed or a chair. Such a problem is named as object servoing.

Instead of selecting feature points arbitrarily from the refer-
ence image and query image as in [14], feature points are se-
lected from the given object. Let \( m_i = [ u_i \ v_i ]^T \) be the \( i \)th feature point of the movable object in the current frame and \( P_i = [ X_i \ Y_i \ Z_i ]^T \) be the coordinate of the corresponding pixel in the current frame. The relationship between \( m_i \) and \( P_i \) is represented as

\[
\begin{bmatrix}
  u_i \\
  v_i
\end{bmatrix} = \pi(P_i) = \begin{bmatrix}
  f_x \frac{X_i}{x_i} + c_x \\
  f_y \frac{Y_i}{y_i} + c_y
\end{bmatrix},
\]

(2)

where \( f_x \) and \( f_y \) are the focal lengths of the camera in pixels, and \( c_x \) and \( c_y \) are the coordinates of the principle point in pixels. For simplicity, a vector \( [ x_i \ y_i ]^T \) is defined as

\[
\begin{bmatrix}
  x_i \\
  y_i
\end{bmatrix} = \begin{bmatrix}
  \frac{X_i}{x_i} \\
  \frac{Y_i}{y_i}
\end{bmatrix} = \begin{bmatrix}
  \frac{u_i-c_x}{f_x} \\
  \frac{v_i-c_y}{f_y}
\end{bmatrix}.
\]

(3)

Similarly, let \( m_i^* = [ u_i^* \ v_i^* ]^T \) be the \( i \)th feature point of the movable object in the goal frame, \( P_i^* = [ X_i^* \ Y_i^* \ Z_i^* ]^T \) be the corresponding coordinate in the goal frame. A vector \( [ x_i^* \ y_i^* ]^T \) is defined as

\[
\begin{bmatrix}
  x_i^* \\
  y_i^*
\end{bmatrix} = \begin{bmatrix}
  \frac{X_i^*}{x_i^*} \\
  \frac{Y_i^*}{y_i^*}
\end{bmatrix} = \begin{bmatrix}
  \frac{u_i^*-c_x}{f_x} \\
  \frac{v_i^*-c_y}{f_y}
\end{bmatrix}.
\]

(4)

To simplify the on-line computation, it is assumed that \( X_i^* \)s are available. They can be determined using any existing method off-line. The relationship between \( P_i \) and \( P_i^* \) is given as

\[
P_i = R P_i^* + T.
\]

(5)

Similar to [14], define \( x_e \) and \( y_e \) as

\[
\begin{bmatrix}
  x_e \\
  y_e
\end{bmatrix} = \begin{bmatrix}
  1 & \sin \theta & \cos \theta \\
  \frac{1}{y_e} & \cos \theta & -\sin \theta
\end{bmatrix} \begin{bmatrix}
  x_i \\
  y_i
\end{bmatrix} - \begin{bmatrix}
  \frac{v}{z_i} \\
  \frac{\omega}{z_i}
\end{bmatrix},
\]

(6)

and \( \theta_e \) as \(-\theta\). It can be derived that

\[
\begin{bmatrix}
  \dot{x}_e \\
  \dot{y}_e \\
  \dot{\theta}_e
\end{bmatrix} = \begin{bmatrix}
  0 & \omega & 0 \\
  -\omega & 0 & 0 \\
  0 & 0 & 0
\end{bmatrix} \begin{bmatrix}
  x_e \\
  y_e \\
  \theta_e
\end{bmatrix} - \begin{bmatrix}
  \frac{v}{z_i} \\
  \frac{\omega}{z_i}
\end{bmatrix},
\]

(7)

where \( v \) and \( \omega \) are the linear velocity and angular velocity of the on-board camera.

Defining a new set of states as

\[
\begin{cases}
  z_0 = -\theta_e = \theta, \\
  z = \begin{bmatrix}
  z_1 \\
  z_2
\end{bmatrix}^T = \begin{bmatrix}
  y_e \\
  -x_e
\end{bmatrix}^T,
\end{cases}
\]

(8)

and a new set of control inputs as [25]

\[
\begin{bmatrix}
  u_0 \\
  u_1
\end{bmatrix} = \begin{bmatrix}
  \frac{\omega}{z_i} - z_1 u_0 \\
  z_1
\end{bmatrix},
\]

(9)

it follows that

\[
\begin{bmatrix}
  \dot{z}_0 \\
  \dot{z}
\end{bmatrix} = \begin{bmatrix}
  u_0 \\
  u_0 A z + B u_1
\end{bmatrix},
\]

(10)

where the matrix \( A \) and the vector \( B \) are

\[
A = \begin{bmatrix}
  0 & 1 \\
  0 & 0
\end{bmatrix}; \quad B = \begin{bmatrix}
  0 \\
  1
\end{bmatrix}^T.
\]

(11)

The object servoing problem is formulated as follows:

Regulate the differential-drive robot to its desired pose by image feedback such that the goal poses of the robot and the movable object satisfy the predefined relative relationship by \( F^* \) regardless of the pose of the movable object, i.e., both \( z_0 \) and \( z \) approach zeros. The linear velocity and angular velocity of the differential-drive robot are zeros when the relative relationship is satisfied, i.e., both \( u_0 \) and \( u_1 \) are zeros when \( z_0 \) and \( z \) are zeros.

The object servoing is different from the existing visual servoing in the sense that the goal pose is fixed in the visual servoing while it is not fixed in the object servoing. The goal pose in the object servoing is determined by the pose of a given object which could be changed.

3 The Proposed Object Servoing

An object servoing scheme will be designed in this section. The proposed scheme consists of an object servoing friendly parking controller and a simple visual motion estimation algorithm. The controller is obtained by integrating a fractional-order controller into the switched controller in [7].

3.1 An Object Servoing Friendly Parking Controller

Similar to [7], it can be shown that the system (10) is asymptotically stabilized by the improved parking controller in Algorithm [7]

In Algorithm [7], \( P_i (i = 1, 2, 3) \) are obtained by solving the following Riccati equation [5]:

\[
A^T P + P A - 2 P B B^T P + (2 \gamma + \kappa_0) P + \kappa_0 L P L = 0,
\]

(16)

where \( \kappa_0 \) is a positive constant. The value of \( \gamma \) is \((\kappa_0 \epsilon + \xi)\) with \( \epsilon(>1) \) and \( \xi \) being two positive constants. Two matrices \( P \) and \( L \) are defined as

\[
P = \begin{bmatrix}
  P_1 \\
  P_2 \\
  P_3
\end{bmatrix}; \quad L = \begin{bmatrix}
  0 & 0 \\
  0 & 1
\end{bmatrix}.
\]

(17)

The invariant set \( \Gamma \) is defined as follows [7]:

\[
\{ [ z_0 \ z^T ] \mid V(z_0, z) < \delta(\kappa_0 \epsilon) \} \text{ or } |z_0| + |z_1| = 0\},
\]

(18)

where \( \delta \) is a positive constant, and the function \( V(z_0, z) \) is defined as

\[
V(z_0, z) = P_1 z_0^2 - 2 P_2 \kappa_0 z_0 z_1 z_2 + P_3 \kappa_0^2 z_1^2 z_2^2.
\]

(19)
Algorithm 1: An improved asymptotic stabilizer of the error system \[10\]

**Inputs:** \(\kappa_0 > 0, \kappa_2 > 0, \epsilon > 1, \xi > 0, \) and \(\delta > 0 \quad [5]\)

**Step 1.** Define the controller \(\gamma = \gamma_0 + \xi, \zeta = 2\gamma + \kappa_0, \) and 
\(\kappa_1 = \frac{2\gamma + 6\kappa_0 \zeta + z^2 + 1}{4}.\)

**Step 2.** Compute three constants \(P_i (i = 1, 2, 3)\) as
\[
\begin{bmatrix}
P_3 \\
P_2 \\
P_1 \\
\end{bmatrix} = \begin{bmatrix}
\kappa_0 + 3\gamma + 6\kappa_0 \zeta + z^2 \\
\kappa_2 - \frac{\kappa_0 + \zeta^2}{2} \\
\frac{\kappa_2}{\sqrt{2}} \zeta \\
\end{bmatrix}.
\]

**Step 3.** Define the controller \(u_0\) as \[7\mid 15\]
\[
u_0 = \begin{cases}
-z_0^2 & \text{if } z_1 = z_2 = 0 \\
\kappa_0 z_0 & \text{otherwise if } [z_0 \ z_1 \ z_2] \in \Gamma, \\
\frac{z_0}{z_1^2} & \text{otherwise}
\end{cases},
\]

\[
p(z_2) = \begin{cases}
z_2; & \text{if } z_2 \neq 0 \\
\text{sign}(z_0 z_1); & \text{otherwise .}
\end{cases}
\]

**Step 4.** Define the controller \(u_1\) as \[7\mid 15\]
\[
u_1 = \begin{cases}
-z_2^2 & \text{if } z_1 = z_0 = 0 \\
-\kappa_2 z_2; & \text{otherwise if } u_0 = 0 \\
-\frac{p_3}{u_0} z_1 - P_3 z_2; & \text{otherwise}
\end{cases}.
\]

Without loss of generality, it is assumed that there are common feature points between the movable object in the first query image and the object in the reference image. A parking controller is object servoing friendly if there usually exist common feature points between the movable object in the subsequent query image and the object in the reference image when the robot is moved under the controller. The controller in Algorithm \[7\] is analyzed as below.

The invariant set \(\Gamma\) includes a set \(S_{s,c}\) which is defined as
\[
S_{s,c} = \{(x_e, y_e, \theta_e) | \theta_e = 0, y_e = 0\},
\]
and the set \(S_{s,c}\) is actually part of the “singularity line” which is determined by the goal pose. It can be easily verified that the set \(S_{s,c}\) is a subset of the following set \(S_s:\)
\[
S_s = \{(x_e, y_e, \theta_e) | \theta_e = 0\},
\]
It is believed in \[5\mid 6\mid 14\] that the system is uncontrollable in the singularity set \(S_s\), and it is challenging to design a controller for asymptotic stabilization of a differential-drive robot when the pose of the robot is in the singular set. Therefore, the set \(S_{s,c}\) is being avoided or escaped by the existing parking control algorithms \[5\mid 6\mid 14\]. On the other hand, the object is well observed if the pose of the robot is in the set \(S_{s,c}\).

Defining a function of \(z_1\) as
\[
\varphi(z_1) = \frac{1}{2} z_1^2, \quad (22)
\]
it can be verified that
\[
\frac{d\varphi(z_1)}{dt} = \begin{cases}
-\kappa_1 z_1^2; & \text{if } z_2 \neq 0 \\
0; & \text{otherwise .}
\end{cases} \quad (23)
\]
This implies that the set \(S_{s,c}\) is approached by the controller in Algorithm \[7\] if the state \([z_0 \ z_1 \ z_2]^T\) is not in the set \(\Gamma\). Therefore, the controller in Algorithm \[7\] is more object servoing friendly than the controller in \[5\mid 6\mid 14\].

### 3.2 A Simple Visual Motion Estimation Algorithm

\(z_i (i = 0, 1, 2\) are used by the controller in Algorithm \[7\] According to the equations \[3\mid 4\mid 5\], both \(R\) and \(T\) will be required to compute \(z_i (i = 0, 1, 2\) for each query image. Once a query image is captured on-line, the feature points of the given object in the reference image will match those feature points in the query image. All matched pairs will be utilized to estimate \(R\) and \(T\) for the query image. It should be pointed out that the exposures of the query image and the reference image could be differently \[19\mid 20\mid 21\]. Their matching could be addressed by using the mapping method in \[22\].

\(\sin \theta, \cos \theta, t_x\) and \(t_y\) are computed by using matched feature points of the given object between each query image and the reference image. For simplicity, \(\sin \theta\) and \(\cos \theta\) are first computed. \(t_x\) and \(t_y\) are then computed separately.

Considering two pairs of matched feature points \((P_i, P_i^\ast)\) and \((P_j, P_j^\ast)\), it can be derived from Equations \[3\mid 4\mid 5\] that
\[
\begin{bmatrix}
x_i \\
y_i
\end{bmatrix} = \begin{bmatrix}
x_i^\ast (x_i^\ast \cos \theta + \sin \theta) + t_x \\
x_i^\ast (\cos \theta - x_i^\ast \sin \theta) + t_y
\end{bmatrix} \quad : \quad i \in \{i, j\}, \quad (24)
\]
It follows that \[14\]
\[
\begin{bmatrix}
a_{ij} \\
b_{ij} \\
c_{ij}
\end{bmatrix} T \quad \text{is computed by}
\]
\[
\begin{bmatrix}
a_{ij} \\
b_{ij} \\
c_{ij}
\end{bmatrix} = \begin{bmatrix}
\frac{y_i}{y_i} (x_i x_j^\ast + 1) - \frac{y_i}{y_j} (x_i^\ast x_j + 1) \\
\frac{y_i}{y_i} (x_i^\ast x_j - x_i) - \frac{y_i}{y_j} (x_i^\ast - x_j) \\
\frac{y_i}{y_i} (x_i - x_j) + (x_i^\ast - x_j^\ast)
\end{bmatrix}. \quad (25)
\]

The optimal values of \(\sin \theta\) and \(\cos \theta\) can be obtained by solving the following optimization problem:
\[
\text{arg min}_{\sin \theta, \cos \theta} \{ \sum_{i,j} (a_{ij} \sin \theta + b_{ij} \cos \theta + c_{ij})^2 \}, \quad (27)
\]
s.t.
\[
\sin^2 \theta + \cos^2 \theta = 1. \quad (28)
\]
Let the cost function of the above optimization problem be denoted as \( \Phi(\sin \theta, \cos \theta) \). The function \( \Phi(\sin \theta, \cos \theta) \) is computed as
\[
\Phi = \sum_{i,j} (a_{ij} \sin \theta + b_{ij} \cos \theta + c_{ij})^2 + \lambda(\sin^2 \theta + \cos^2 \theta - 1).
\] (29)

It can be derived from the partial derivatives that the optimal values of \( \sin \theta \) and \( \cos \theta \) are
\[
\begin{bmatrix}
\sin \theta \\
\cos \theta
\end{bmatrix} = \begin{bmatrix}
\frac{(\tilde{a}_3 + \lambda)\tilde{b}_1 - \tilde{a}_2 \tilde{b}_2}{(\tilde{a}_1 + \lambda)(\tilde{a}_3 + \lambda) - \tilde{a}_2^2} \\
\frac{(\tilde{a}_1 + \lambda)\tilde{b}_2 - \tilde{a}_3 \tilde{b}_1}{(\tilde{a}_1 + \lambda)(\tilde{a}_3 + \lambda) - \tilde{a}_2^2}
\end{bmatrix},
\] (30)
where \( \begin{bmatrix} \tilde{a}_1 & \tilde{a}_2 & \tilde{a}_3 \end{bmatrix}^T \) and \( \begin{bmatrix} \tilde{b}_1 & \tilde{b}_2 \end{bmatrix}^T \) are computed by
\[
\begin{bmatrix}
\tilde{a}_1 \\
\tilde{a}_2 \\
\tilde{a}_3
\end{bmatrix} = \begin{bmatrix}
\sum_{i,j} a_{ij}^2_i \\
\sum_{i,j} a_{ij} b_{ij} \\
\sum_{i,j} b_{ij}^2_i
\end{bmatrix},
\] (31)
\[
\begin{bmatrix}
\tilde{b}_1 \\
\tilde{b}_2
\end{bmatrix} = \begin{bmatrix}
-\sum_{i,j} a_{ij} c_{ij} \\
-\sum_{i,j} b_{ij} c_{ij}
\end{bmatrix}.
\] (32)

Combining Equations (28) and (30), it can be obtained that
\[
\lambda^4 + 2c_1 \lambda^3 + c_2 \lambda^2 + 2c_3 \lambda + c_4 = 0,
\] (33)
where \( \begin{bmatrix} c_1 & c_2 & c_3 & c_4 \end{bmatrix}^T \) is computed as
\[
\begin{bmatrix}
\tilde{a}_1 + \tilde{a}_3 \\
(\tilde{a}_1 + \tilde{a}_3)^2 - \tilde{b}_1^2 - \tilde{b}_2^2 + 2(\tilde{a}_1 \tilde{a}_3 - \tilde{a}_2^2) \\
(\tilde{a}_1 + \tilde{a}_3)(\tilde{a}_1 \tilde{a}_3 - \tilde{a}_2^2) + 2\tilde{a}_2 \tilde{b}_1 \tilde{b}_2 - \tilde{a}_3 \tilde{b}_1^2 - \tilde{a}_1 \tilde{b}_2^2 \\
(\tilde{a}_1 \tilde{a}_3 - \tilde{a}_2^2)^2 - (\tilde{a}_3 \tilde{b}_1 - \tilde{a}_2 \tilde{b}_2)^2 - (\tilde{a}_1 \tilde{b}_2 - \tilde{a}_2 \tilde{b}_1)^2
\end{bmatrix}.
\] (34)

Once \( \lambda \) is obtained from Equation (33), \( \sin \theta \) and \( \cos \theta \) can be computed using Equation (30).

When \( t_x \) and \( t_y \) are computed, the 3D information of feature points in one of the reference and query images is required. To reduce the on-line complexity, the reference image is supposed to include the 3D information of the feature points while the query image only includes the 2D information of the feature points.

Using Equation (24), \( t_x \) and \( t_y \) are obtained via solving the following optimization problem:
\[
\arg \min_{t_x, t_y} \left\{ \sum_i \left[ (d_i - x_{i,t})^2 + (e_i + x_i t_x - t_y)^2 \right] \right\},
\] (35)

where \( d_i \) and \( e_i \) are
\[
\begin{bmatrix}
d_i \\
e_i
\end{bmatrix} = \begin{bmatrix}
X_i^* \left( \frac{y_i}{y_i^*} - (\cos \theta - x_i^* \sin \theta) \right) \\
X_i^* \left( (x_i - x_i^*) \cos \theta - (x_i x_i^* + 1) \sin \theta \right)
\end{bmatrix}.
\] (36)

Their optimal values are computed as
\[
\begin{bmatrix}
t_x \\
t_y
\end{bmatrix} = \frac{N \sum_i (d_i - x_i t_x + \sum_i e_i)}{N \sum_i (x_i t_x + \sum_i e_i)}.
\] (37)

Compared with a possible alternative by estimating the pose of the movable object on-line \[23\], the on-line computational cost of the proposed motion estimation method is much lower.

4 Simulation Results

In this section, simulation results are provided to illustrate the proposed controller is more object servoing friendly. The simulation results for the controller in \[5\] are also provided to compare the two controllers. The parameters of the existing controller are selected according to the recommendation in the original paper \[5\]. The controller parameters are summarised in Table 1. Four sets of initial and target poses are listed in Table 2.

![Figure 2](image)

**Figure 2:** Curves of \( |z_0| + |z_1| \). (a) and (c) by the controller in \[5\]; (b) and (d) by the controller in this paper.

| Controller            | \( \beta \) | \( \kappa_0 \) | \( \alpha \) | \( \kappa_2 \) | \( \epsilon \) | \( \xi \) | \( \delta \) |
|-----------------------|-----------|-------------|-----------|-------------|-------------|------|--------|
| Controller in \[5\]   | 2         | 0.1         | N.A.      | 2.05        | 0.01        | 1/1024| 25     |
| Proposed controller   | N.A.      | 0.1         | 0.01      | 1/1024      | 2.25        | 1/1024| 25     |

| Case       | Target pose |
|------------|-------------|
| Initial pose | (0, 0, \( \pi/6 \)) | (0, 0, \( \pi/4 \)) | (5, 5, \( \pi/6 \)) | (5, 5, 0) |
| Target pose | (5, 5, 0) | (5, 5, 0) | (16, 6, \( \pi/6 \)) | (16, 6, \( \pi/6 \)) |

For Case 1, it is demonstrated by Fig. 3 (a) that the singularity set is indeed being escaped from by using the controller in \[5\]. When the proposed controller is employed, it can be
Figure 3: Simulation trajectories (left) and control outputs (right) using the proposed controller and the controller in [5] without kinematic constraints. (a) and (b): Trajectory and output by the controller in [5] for Case 1; (c) and (d): Trajectory and output by the proposed controller for Case 1; (e) and (f): Trajectory and output by the controller in [5] for Case 2; (g) and (h): Trajectory and output by the proposed controller for Case 2.

Figure 4: Simulation trajectories (left) and control outputs (right) using the proposed controller and the controller in [5] with kinematic constraints. (a) and (b): Trajectory and output by the controller in [5] for Case 3; (c) and (d): Trajectory and output by the proposed controller for Case 3; (e) and (f): Trajectory and output by the controller in [5] for Case 4; (g) and (h): Trajectory and output by the proposed controller for Case 4.

observed from Fig. 3(c) that the robot is kept in the set $S_{s,c}$, resulting in much smoother and shorter trajectory. The linear velocity and angular velocity required by the proposed controller are much smaller than those required by the existing controller as well. For Case 2, the initial orientation is set to denoted by the green “arrow”, while the target pose remains unchanged in Fig. 3(g), the proposed controller is able to stabilize the robot at target pose with less control effort compared to the controller in [5]. In order to apply robot in some practical situations, for Case 3 and Case 4, the linear and angular velocity of the robot are set to be bounded by 1 m/s and 1 rad/s, respectively. The simulation results in Fig. 4 are consistent with the previous case. It can be observed that the robot is able to approach the target poses using both controllers. However, it is illustrated by Fig. 4(c) and (g) that the proposed controller not only results in a shorter object servoing trajectory, but also doesn’t cause oscillating behaviours as shown in Fig. 4(b) and (f). In addition, it is worth noting that a parallel object servoing task can be efficiently performed by the proposed controller in Case 3. For Case 1 and Case 2, the curves of $|z_0| + |z_1|$ are shown in Fig. 4. Normally, there are more overlapping area between a query image and the reference image if the value of $(|z_0| + |z_1|)$ is smaller. The value of $(|z_0| + |z_1|)$ decreases sharply to a small value using the
proposed controller while it increases first before decreasing using the controller in [5]. The simulation results indicate that the proposed controller approaches the controllable “singularity line”, while the controller in [5] tries to escape from the singularity set during the parking process. As a result, the proposed controller is much more object servoing friendly than the controller in [5].

5 Conclusion and Future Works

A novel object servoing scheme has been designed in this paper for differential-drive robots using visual based motion estimation and an object servoing friendly parking controller. The proposed scheme can be adopted to park a differential-drive robot at a predefined relative pose to a movable object. Due to the low on-line computational cost of the proposed scheme, it could be adopted for the last mile delivery of mobile robots to movable objects. It should be pointed out that the proposed scheme might not perform well in very cluttered environments.

REFERENCES

[1] F. Chaumette and S. Hutchinson, “Visual servo control—Part I: Basic approaches,” IEEE Robot. Autom. Mag., vol. 13, no. 4, pp. 82-90, Dec. 2006.
[2] F. Chaumette and S. Hutchinson, “Visual servo control—Part II: Advanced approaches,” IEEE Robot. Autom. Mag., vol. 14, no. 1, pp. 109-118, Mar. 2007.
[3] R. W. Brockett, “Asymptotic stability and feedback stabilization, in differential geometric control theory,” R. W. Brockett, R. S. Millman, and H. J. Sussmann eds., Boston, MA: Birkhauser, 1983, pp.181-191.
[4] Y. P. Tian and S. H. Li, “Smooth exponential stabilization of nonholonomic systems via time-varying feedback,” Automatica, vol. 38, no. 7, pp. 1139–1146, Jul. 2002.
[5] A. Marchand and M. Alamir, “Discontinuous exponential stabilization of chained form systems,” Automatica, vol. 39, no. 2, pp.343-348, Feb. 2003.
[6] J. Luo and P. Tsiotras, “Control design for chained-form systems with bounded inputs,” Systems & Control Letters, vol. 39, no. 2, pp. 123–131, Feb. 2000.
[7] Z. G. Li, W. C. Gao, C. Z. Goh, M. L. Yuan, E. K. Teoh, and Q. Y. Ren, “Asymptotic stabilization of nonholonomic robots leveraging singularity,” IEEE Robotics and Automation Letters, vol. 4, no. 1, pp. 41-48, Jan. 2019.
[8] P. Murrieri, D. Fontanella, and A. Bicchi, “A hybrid-control approach to the parking problem of a wheeled vehicle using limited view-angle visual feedback,” Int. J. Robot. Res., vol. 23, no. 4-5, pp. 437-448, 2004.
[9] Z. G. Li, Y. C. Soh, and C. Y. Wen, “Robust stability of a class of hybrid nonlinear systems,” IEEE Trans. on Automatic Control, vol. 46, no. 6, pp.897-903, Jun. 2001.
[10] Z. G. Li, Y. C. Soh, and C. Y. Wen, Switched and impulsive systems: Analysis, design and applications, vol. 313. Springer Science & Business Media.
[11] K. Li, Y. C. Soh, and Z. G. Li, “Chaotic cryptosystem with high sensitivity to parameter mismatch,” IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications, vol. 50, no. 4, pp. 579-583, Apr. 2003

[12] N. Gans and S. Hutchinson, “A stable vision-based control scheme for nonholonomic vehicles to keep a landmark in the field of view,” in Proc. IEEE Int. Conf. Robot. Autom., pp. 2196–2201, 2007.
[13] G. Lopez-Nicolas, S. Bhattacharya, J. J. Guerrero, C. Sagues, and S. Hutchinson, “Switched homography-based visual control of differential drive vehicles with field-of-view constraints,” in Proc. IEEE Int. Conf. Robot. Autom., pp. 4238–4244, 2007.
[14] X. Zhang, Y. Fang, and X. Liu, “Motion-estimation-based visual servoing of nonholonomic mobile robots,” IEEE Trans. On Robotics, vol. 27, no. 6, pp. 1167-1175, Dec. 2011.
[15] Y. Q. Chen, I. Petras, and D. Xue, “Fractional order control - a tutorial,” in American Control Conference, pp.1397-1411, 2009.
[16] K. He, G. Gkioxari, P. Doll’ar, and R. Girshick, “Mask R-CNN,” arXiv preprint arXiv:1703.06870 2017.
[17] J. Vidal, C. Lin, X. LLado, and R. Marti, “A method for 6D pose estimation of free-form rigid objects using point pair features on range data,” Sensors, pp. 1-20, 2678, Aug. 2018.
[18] Z. Li, G. Wang, and X. Ji, “CDPN: coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation,” in ICCV, pp. 1-8, 2019.
[19] J. Zheng, Z. Li, Z. Zhu, S. Wu, and S. Rahardja, “Hybrid patching for a sequence of differently exposed images with moving objects,” IEEE Trans. on Image Processing, vol. 22, no. 12, pp. 5190-5201, Dec. 2013.
[20] G. Li, C. Y. Wen, Z. G. Li, F. Yang, and K. Z. Mao, “Model based online learning with kernels,” IEEE Trans. on Neural Networks and Learning Systems, vol. 24, no. 3, pp. 356-369, Mar. 2013.
[21] F. Kou, Z. G. Li, W. H. Chen, and C. Y. Wen, “Multi-scale exposure fusion via gradient domain guided image filtering,” in IEEE 2017 International Conference on Multimedia and Expo, Hong Kong, Jul. 10-14, 2017.
[22] J. Jiang, Z. G. Li, S. L. Xie, S. Q. Wu, and L. C. Zeng, “Robust alignment of multi-exposed images with saturated regions,” IEEE Access, vol. 8, pp. 221689-221699, Dec. 2020.
[23] Y. Xiang, T. Schmidt, V. Naravanan, and D. Fox, “PoseCNN: a convolutional neural network for 6D object pose estimation in cluttered scenes”, in Robotics: Science and Systems 2018, pp. 1-10, USA, 2018.
[24] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, “Monte Carlo localization: efficient position estimation for mobile robots,” in Proceedings of the sixteenth national conference on Artificial intelligence, pp. 343-349, USA, Jul. 1999.
[25] G. Oriolo, A. De Luca, and M. Vendittelli, “WMR control via dynamic feedback linearization: design, implementation, and experimental validation,” IEEE Trans. Control Syst. Technol., vol. 10, no. 6, pp. 835-852, Nov. 2002.