Abstract—Traditional image based visual servoing approaches require feature extraction and tracking during the servoing process. Feature extraction and tracking, however, can be quite difficult in some cases. To eliminate these requirements, this paper presents a visual servoing approach which performs control directly using the intensity information in the image. Considering the image as a set, the approach formulates the servoing problem in the space of sets, which is called non-vector space in this paper. The control theory in this non-vector space is used to formulate the stabilization problem. A controller is designed to stabilize the system around a desired image set. Three types of experiments are carried out on a redundant robotic manipulator to validate the controller. The experimental results verify the correctness of the controller. The approach presented in this paper can be applied to robotic manipulation and navigation.

I. INTRODUCTION

Visual servoing uses visual information to control a mechanical system to move from an initial position to a desired position. For image based control, this is accomplished by moving the imaging device in such a way that the image will converge to a predefined desired image. This method requires extraction and tracking of geometric features in the image [1]. Example features include points, lines, or image moments, etc [2]. In reality, however, reliable feature extraction and tracking are quite difficult. In fact, it is one of the most difficult problems in computer vision [3].

To avoid feature extraction and tracking, a direct visual servoing approach is emerging in recent years. In this method, the intensities in grey scale images are directly used for visual servoing. Several different variations for this method have been proposed. According to how the error is defined, they can be grouped into three categories.

In the first category, the whole image is transformed into a low dimensional space, and controllers are designed to make the error in this new space converge to zero. Deguchi performs principle component analysis to transform the image to a low dimensional eigenspace. Although no explicit feature extraction is needed, this method requires an off-line eigenspace decomposition of the image [4]. Kallem et al. project the image to a low dimensional kernel measurement via a kernel projection, and controllers for several subsets of three dimensional rigid motion are designed to minimize the error between the current and desired kernel measurements [5].

Different from the first category, the methods in the second category define the error as the sum of square differences (SSD) between the intensities vectors from the current and desired images. Controllers are then designed to minimize the SSD. Collewet and Marchand derive the interaction matrix using the Phong illumination model and obtain an optimization based control law to minimize the SSD [6]. With a similar idea, Han and Censi et al. investigate a bi-plausible method to stabilize the pose of a helicopter with intensity information from the image [7]. Although they also try to minimize the SSD, the sensing model is spherical, and the dynamics model is used instead of traditional kinematics model in visual servoing.

The final approach exploits the mutual information, an idea from information theory, between the goal and current image based on the intensity information. Then the mutual information is maximized so that the current image will be aligned with the desired image. Dame and Marchand initiate this approach, and experimental results demonstrate its robustness to illumination variance and depth approximations [8].

In this paper, a new direct visual servoing method is presented. Considering the images as sets, we can obtain a controller to steer the current image set to the desired image set. Since the derivation is performed in the space of sets, and the linear structure of the vector space does not hold any more, we call this approach non-vector space visual servoing (NSVS). Note that this method is fundamentally different from above approaches because the error is defined between sets instead of vectors. Consequently, a set of new tools is needed to deal with set dynamics [9].

This method is first proposed by Doyen [10]. Recently, we validate the approach with images obtained by atomic force microscopy [11]. We have also showed that it still works even only partial image information is known [12]. Our previous work, however, only deals with the cases when the imaging

Fig. 1. The schematic for non-vector space visual servoing approach

In this paper, a new direct visual servoing method is presented. Considering the images as sets, we can obtain a controller to steer the current image set to the desired image set. Since the derivation is performed in the space of sets, and the linear structure of the vector space does not hold any more, we call this approach non-vector space visual servoing (NSVS). Note that this method is fundamentally different from above approaches because the error is defined between sets instead of vectors. Consequently, a set of new tools is needed to deal with set dynamics [9].
The subsets in $Y$ includes all the points in $R$ to be defined. A tube distance from a set $L$ is a Lipschitz function. Denote the set of all such functions as $\Phi \in \text{BL}(E, \mathbb{R}^n)$. Then the transition, a special tube induced by $\varphi \in \text{BL}(E, \mathbb{R}^n)$, is defined as:

$$T_\varphi(t, K_0) = \{x(t) : x = \varphi(x), x(0) \in K_0\}$$

where $K_0 \subset \mathbb{R}^n$ is a set containing all the starting points for $x = \varphi(x)$. Based on the transition and the Hausdorff distance, a special function $\varphi$ satisfying the following condition

$$\lim_{\Delta t \to 0^+} \frac{1}{\Delta t} d h(K(t + \Delta t), T_\varphi(\Delta t, K(t))) = 0$$

can be considered as the derivative of a set. In the above definition, the first set is the set of tube at time $t + \Delta t$. The other set is the transition set starting from $K(t)$ after time $\Delta t$. The Hausdorff distance between these two sets divided by the time interval $\Delta t$ will go to zero as $\Delta t$ approaches zero. The above definition can be considered as an analogous to the first order approximation for the curve velocity in vector space [13].

The mutation of a tube $K(t)$ is the set of the $\varphi$ satisfying Eq. (3), that is:

$$\dot{K}(t) = \{\varphi(x) : \lim_{\Delta t \to 0^+} \frac{1}{\Delta t} d h(K(t + \Delta t), T_\varphi(\Delta t, K(t))) = 0\}$$

With the definition of mutation, the dynamics of set can be defined by the mutation equation as:

$$\varphi(x) \in \dot{K}(t)$$

To add control input to the mutation equation, we can consider the mapping $\varphi : E \times U \to \text{BL}(E, \mathbb{R}^n)$ where $U$ is the set of all possible controls $u$. Then the controlled mutation equation is:

$$\varphi(x(t), u(t)) \in \dot{K}(t) \quad \text{with} \quad u(t) = \gamma(K(t))$$

where the feedback map $\gamma : \mathcal{P}(\mathbb{R}^n) \to U$ generates a control input from the current set $K(t)$.

### B. Stabilizing Controller in the Non-vector Space

In the non-vector space, the stabilization problem can be formulated as: Given a desired set $\mathcal{K}$ and an initial set $K(0)$ in the vicinity of $\mathcal{K}$, design a controller $u(t) = \gamma(K(t))$ based on the current set $K(t)$ such that $d h(K(t), \mathcal{K}) \to 0$ as $t \to \infty$.

For a special system when $\varphi(x(t), u(t))$ in the controlled mutation Eq. (6) is linear in $u$, a controller can be designed. In this case, let $\varphi(x, u) = L(x)u$ with $u \in \mathbb{R}^n$ and $L(x) \in \mathbb{R}^{m \times n}$. Note that the time $t$ is omitted for clear presentation. The stabilizing controller is obtained by extending the Lyapunov theory into the non-vector space [10]. The following Lyapunov function candidate can be used:

$$V(K) = \int_0^\infty d K(x) dx + \int_0^\infty d K(x) dx$$

Note that $V(K)$ is a function of a set, and its derivative requires special techniques. The detail derivation can be found in [14]. With the Lyapunov function $V(K)$, we have the following theorem:
Theorem 1 [10], [11] The system \( L(x)u \in \hat{K}(t) \) with \( x \in \mathbb{R}^m, L(x) \in \mathbb{R}^{m \times n}, u \in \mathbb{R}^n \), and \( K \subset \mathbb{R}^m \), can be locally exponentially stabilized at \( \hat{K} \) by the following controller:

\[
u(t) = \gamma(K) = -\alpha \frac{D^T(K)D(K)V(K)}{D^T(K)V(K)}(8)\]

where \( \alpha > 0 \) is a gain factor and \( D(K) \) is a row vector defined by:

\[
D(K) = \frac{1}{2} \int_{\hat{K}} d^2K(x)[(\sum_{i=1}^{m} \partial L_i/\partial x_i)]dx + \int_{\hat{K}} (x-P_{K}(x))^TL(x)dx
- \int_{\hat{K}} (x-P_{K}(x))^TL(P_{K}(x))dx
\]

where \( L_i(i=1,2,\ldots,m) \) are the \( m \) row vectors in matrix \( L \), and \( \partial L_i/\partial x_i \) is also a row vector with the same dimension. Note that the time \( t \) is also omitted for clear presentation. The controller in Eq. (8) can solve the problem proposed at the beginning of this subsection. The proof of this theorem is given in [10].

III. VISUAL SERVOING IN THE NON-VARIABLE SPACE

The visual servoing problem can be modeled by the special system in the above theorem, and the controller can be readily applied. In fact, for servoing with grey scale images, each pixel can be represented by a three dimensional vector \( x = [x_1, x_2, x_3]^T \) where \( x_1 \) and \( x_2 \) are the pixel indices, and \( x_3 \) the pixel intensity. For a general visual servoing problem, the control input is the camera’s spatial velocity. Therefore, the control input \( u(t) \) has three translational components and three rotational components, which can be represented by a row vector \( u(t) = [v_x, v_y, v_z, \omega_x, \omega_y, \omega_z]^T \).

The system in theorem 1 is determined by \( \phi(x) \), which is further determined by the relationship between \( u(t) \) and \( x(t) \). The perspective projection sensing model in computer vision can be used to derive such a relationship. Under constant lighting condition, \( x_3 \) will be a constant for each pixel; therefore, \( x_3 = 0 \). With a unit camera focal length, a 3D point with coordinates \( P = [p_x, p_y, p_z]^T \) in the camera frame will be projected to the image plane with coordinates:

\[
x_1 = p_x/p_z \quad x_2 = p_y/p_z \quad (10)
\]

Based on these equations, the relation between \( u(t) \) and \( x(t) \) can be obtained as:

\[
\dot{x}(t) = L(x(t))u(t) \quad (11)
\]

where

\[
L = \begin{bmatrix}
-1/p_z & 0 & x_1/p_z & x_1 x_2 & -(1 + x_1^2) & x_2 \\
0 & -1/p_z & x_2/p_z & x_2 x_1 & 1 + x_2^2 & -x_1 x_2 & -x_1 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

Note that first two rows are the same as the interaction matrix in visual servoing [15]. In Eq. (11), \( \dot{x}(t) = \phi(x(t), u(t)) = L(x(t))u(t) \) is linear in \( u \); therefore, the controller in Eq. (8) can be directly applied. From the expression of \( L \), we have:

\[
\sum_{i=1}^3 \frac{\partial L_i}{\partial x_i} = \begin{bmatrix} 0 & 0 & 2/p_z & 3 x_2 & -3 x_1 & 0 \end{bmatrix}
\]

The preliminary experiments for three cases will be carried out to verify the controller in the experimental part. For easy implementation, the simplified controllers for the three cases will be obtained.

A. 2D Translation

In this case, the camera can only translate in the plane perpendicular to its optical axis. Since the depth \( p_z \) is a constant, we can assume it to be one. Therefore, \( L \) is a constant matrix, and the row vector \( D(K) \) can be simplified to:

\[
D(K) = [\int_{\hat{K}} (x-P_{K}(x))^TL(x)dx - \int_{\hat{K}} (x-P_{K}(x))^TLP_{K}(x))]\quad (12)
\]

with

\[
L = \begin{bmatrix}
-1 & 0 \\
0 & -1 \\
0 & 0
\end{bmatrix}
\]

From the formula, we see that the last element (intensity value) in \( x-P_{K}(x) \) or \( P_{K}(x) \) plays no role in the computation of \( D(K) \) since the last row of \( L \) is zero. The intensity value, however, can effect the computation of \( P_{K}(x) \) or \( P_{K}(x) \) because the distance is computed based on the vector \( x = [x_1, x_2, x_3]^T \).

B. 3D Translation

In this case, the camera can only perform the translational motion. The row vector \( D(K) \) can be simplified to:

\[
D(K) = \int_{\hat{K}} \frac{1}{p_z} [0, 0, d^2K(x)]dx + \int_{\hat{K}} \frac{1}{p_z} (x-P_{K}(x))^TL(x)dx
- \int_{\hat{K}} \frac{1}{p_z} (x-P_{K}(x))^TLP_{K}(x)dx
\]

with

\[
L(x) = \begin{bmatrix}
-1 & 0 & x_1 \\
0 & -1 & x_2 \\
0 & 0 & 0
\end{bmatrix}
\]

In general, \( p_z \) varies with different \( x \) in the set \( K \). A special case is when the observed object is planar, then \( p_z \) will be the same for all \( x \in K \), and it can be taken out from the integral. In this case, although \( p_z \) will change during the translation along the optical axis, it only effects the magnitude of \( u(t) \). Therefore, we can assume \( p_z \) to be one.

C. SE(2) Motion

In this case, the camera has three degree-of-freedom (DOF). In addition to 2D translational movement, it can also rotate about its optical axis. The row vector \( D(K) \) can be simplified to:

\[
D(K) = \int_{\hat{K}} (x-P_{K}(x))^TL(x)dx - \int_{\hat{K}} (x-P_{K}(x))^TLP_{K}(x)dx
\]

with

\[
L(x) = \begin{bmatrix}
-1/p_z & 0 & x_2 \\
0 & -1/p_z & -x_1 \\
0 & 0 & 0
\end{bmatrix}
\]

Note that \( p_z \) will be a constant if a planar object is used.
IV. ROBOT CONTROLLER DESIGN

To validate the NSVS approach, experiments with robotic manipulator are conducted. The detail experimental framework is shown in Fig. 2, which is the general look-and-move structure [1]. First of all, the non-vector space controller generates a camera velocity based on the current image from the camera and the desired image. This velocity is resolved into the joint speeds via the inverse velocity kinematics. Then the robot executes this speed through the joint controller until a new joint speed command is received. Note that we use an eye-in-hand configuration with the camera attached to the manipulator’s end-effector.

The key step in Fig. 2 is the inverse velocity kinematics, which is well discussed in standard robotics textbooks for a six DOF manipulator [16]. Nevertheless, we will use a seven DOF manipulator to provide flexibility and capacity by its redundancy. In this case, the redundancy resolution requires extra efforts which will be briefly described in this section.

The LWA3 redundant manipulator with seven revolute joints from Schunk is used for experiment. In order to resolve the redundancy, an extra variable is introduced. It is named arm angle and denoted by \( \phi \). It represents the configuration of the plane constructed by the elbow-shoulder link and elbow-wrist link with respect to the shoulder-wrist axis [17]. Let \( \xi_{ce} \in \mathbb{R}^6 \) be vector for the three linear and three angular velocities of the end-effector. Let \( \theta \in \mathbb{R}^7 \) denote the seven joint angles of the manipulator. Then \( \xi_{ce} \) and \( \dot{\theta} \) has the following relationship:

\[
\xi_{ce} = J_{ce}(\theta) \dot{\theta} \tag{15}
\]

where \( J_{ce}(\theta) \in \mathbb{R}^{6 \times 7} \) is the Jacobian matrix of the end-effector. The arm angle \( \phi \) introduced earlier is related to the seven joint angles by:

\[
\phi = J_{\phi}(\theta) \dot{\theta} \tag{16}
\]

where \( J_{\phi}(\theta) \in \mathbb{R}^{1 \times 7} \) can be considered as the Jacobian matrix of the arm angle.

Eqs. (15) and (16) can be combined together to obtain:

\[
Y = \begin{bmatrix} \xi_{ce} \\ \phi \end{bmatrix} = \begin{bmatrix} J_{ce}(\theta) \\ J_{\phi}(\theta) \end{bmatrix} \dot{\theta} = J(\theta) \dot{\theta} \tag{17}
\]

where \( J(\theta) \in \mathbb{R}^{7 \times 7} \) is the augmented Jacobian matrix of the full configuration. Since \( J(\theta) \) is a square matrix, a unique velocity for the seven joints can be obtained if \( J(\theta) \) is nonsingular. In this way, the redundancy problem is solved.

With above redundancy resolution, we can obtain the joint speed given the end-effector’s velocity in the end-effector frame. But the output of the non-vector space controller is the velocity of the camera in the camera frame. Therefore, the transformation from the camera frame to the end-effector frame should be derived. Denote the camera velocity in the camera frame with \( \xi_{c} \in \mathbb{R}^6 \). Then we have

\[
\xi_{ce} = J_{ce} \xi_{c} \tag{18}
\]

where \( J_{ce} \in \mathbb{R}^{6 \times 6} \) is the transformation matrix which can be obtained once the spatial relationship between the camera and end-effector is known [16]. From Eqs. (17) and (18), the relationship between the joint velocity and the camera velocity is:

\[
\dot{\theta} = J^{-1}(\theta) \dot{\theta} \tag{19}
\]

where \( J(\theta) \in \mathbb{R}^{7 \times 7} \) is the augmented Jacobian matrix of the new full configuration. Because the manipulator has seven DOF, the three position and three orientation movements of the camera are all achievable. Therefore, we can let camera’s velocity be the output of the non-vector based controller:

\[
\xi_{c} = u(t) \tag{20}
\]

To achieve this output, the joint velocities of the manipulator can be obtained by the inverse velocity kinematics of Eq. (19):

\[
\dot{\theta} = J^{-1}(\theta) \dot{\theta} \tag{21}
\]

where the velocities of the camera are obtained from Eq. (20). The velocity of the arm angle is determined by the online sensors for obstacle avoidance of the manipulator links [18]. Then the joint velocities are executed by the joint motors based on joint motor speed controllers.

V. EXPERIMENTAL RESULTS

To verify the NSVS controller, experiments are conducted on the LWA3 redundant manipulator. Note that the controller can be verified using any manipulator with six DOF, but we use the LWA3 for future application such as obstacle avoidance during the servoing process. Therefore, the experiments are focused on the controller verification instead of redundancy resolution.
The experimental setup is shown in Fig. 3, where an ordinary CMOS camera is rigidly attached to the end-effector. As our preliminary experiment, a planar object with a rectangle, a circle, and a triangle printed on a white paper is placed on the ground. Note that although planar geometric shapes are used, no image feature extraction and tracking are performed in the experiments.

![Experimental setup to verify the NSVS](image)

As the quality of the CMOS camera is poor, the image noise can be quite large. For example, with two consecutive images obtained with a still camera looking at a still environment, the variation of image intensities may be up to 12%. To address this issue, we convert grey scale images to binary images for our experiment. A $120 \times 160$ grey scale image is obtained, then it is converted to a binary image to retain the geometric shapes. The camera is calibrated, and the following intrinsic parameters are found: focal lengths in pixels $f_x = f_y = 204$ and principle points in pixels $u_r = 48$, and $u_c = 92$.

Experiments for the three subsets of rigid motion discussed in section III are carried out. The general process for the experiments is as follows. First of all, the manipulator is moved to a desired position, where a desired image is acquired. The desired position is obtained using the manipulator’s forward kinematics. Then the manipulator is moved to some initial position, where the initial image is recorded. Then the manipulator will start moving according to the framework in Fig. 2. During the movement, the manipulator end-effector’s positions are recorded with 5Hz frequency. In this way, the error trajectory can be obtained after the experiment.

A. 2D Translation

For this experiment, the manipulator is placed at a constant height from the planar object, and it can only move in two translational directions in the plane parallel to the planar object. Four experiments are performed, and one of the results is shown in Fig 4. The desired and initial images have the geometric shapes at two corners in the field of view. The errors in both directions with respect to time are shown in Fig. 4(c). The results show that the error is almost zero after ten seconds. The Hausdorff distance with respect to iterations is shown in Fig. 4(d).

![Experimental results for the 2D translation](image)

The error results in both directions for the four experiments are listed in table I. The average error is also computed, which shows that the error in $x$ direction is larger than that in $y$ direction. Since the manipulator is placed on a mobile base for mobile manipulation [19], [20], this error is acceptable for future use.

<table>
<thead>
<tr>
<th>Trial No.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Final Error $x$ (mm)</td>
<td>2.0</td>
<td>4.5</td>
<td>1.1</td>
<td>2.3</td>
<td>2.5</td>
</tr>
<tr>
<td>$y$ (mm)</td>
<td>0.5</td>
<td>0.2</td>
<td>1.4</td>
<td>1.0</td>
<td>0.8</td>
</tr>
</tbody>
</table>

B. 3D Translation

In this experiment, the manipulator is allowed to move in all the three translational directions. Four experiments are carried out, and the detail of one experiment is shown in Fig. 5. The meaning for each figure is the same to the 2D translation case. As discussed in section III, we can use a constant depth $p_z$ in the controller for planar objects. From the results, we see that the controller can make the manipulator move to the desired position.

![Experimental results for the 3D translation](image)
The final errors in three directions for the four experiments are shown in table II. The average errors are also computed, and the results show performance similar to the 2D translation case.

C. SE(2) Motion

![Desired Image](image1.png) ![Initial Image](image2.png)

![Error in the task space](image3.png) ![Haussdorff distance](image4.png)

Fig. 6. Experimental results for the SE(2) motion

In this experiment, the manipulator can perform the 2D translation and a rotation about its optical axis. Four experiments are conducted, and one result is shown in Fig. 6. Note that the rotation error is also plotted in Fig. 6(c) with degree as the unit. From the plot, there are some overshoots for the system, but the controller can still stabilize the system at the desired position.

The final errors for the four experiments are shown in table III, where the unit for the rotation is degree. From the table, we see that the error in rotation is quite small.

**VI. CONCLUSIONS**

A new visual servoing approach based on the image intensity information is proposed in this paper. This approach considers the images as sets and formulates dynamics in the space of sets. The control theory in the non-vector space is used to obtain the dynamics model. Based on the model, a stabilizing controller is designed to regulate the Hausdorff distance between the current and the desired image sets. Experiments are conducted on a robotic manipulator for three subsets of the rigid motions: 2D translational motion, 3D translational motion, and SE(2) motion. Experimental results show that the controller can steer the manipulator to the desired position, and the error is acceptable for robotic manipulation purposes. Future work will be focused on the implementation of the SE(3) motion and the applications to manipulation or navigation.

**REFERENCES**


