An Image Based Visual Control Law for a Differential Drive Mobile Robot

Indrazno Siradjuddin, Indah Agustien Siradjuddin, and Supriatna Adhisuwignjo

This paper presents the development of an Image Based Visual Servoing control law for a differential drive mobile robot navigation using single camera attached on the robot platform. Four points image features have been used to compute the actuation control signals: the angular velocities of the right wheel and the left wheel. The actuation control signals move the mobile robot at the desired position such that the error vector in the image space has been minimised. The stability of the proposed IBVS control law has been validated in the sense of the Lyapunov stability. Simulations and real-time experiments have been carried out to verify the performances of the proposed control algorithm. Visual servoing platform (ViSP) libraries have been used to develop the simulation program. Real-time experiments have been conducted using a differential drive mobile robot where a Beaglebone Black Board was used as the main hardware controller.

***Index Terms*—Visual servoing, differential drive mobile robot, Beaglebone Black, robotics**

I. Introduction

DEAD reckoning control strategy is a popular method for an autonomous robot navigation [1], [2]. In the case of a mobile robot control, dead reckoning relies on odometri sensors that measure the number of rotations of a robot wheel. Using this technique, the position and the velocity of a mobile robot can be estimated. However, such method subject to estimation error due to wheelslip and discrepancy between the kinematics model and the real robot kinematics [3]. Alternatively, the use vision sensor is a promising method to improve the robot navigation capabilities either in a single or collaborative robot tasks [4], this technique is also known as visual servoing method. Visual servoing methods use the feedback visual information to provide a reactive motion behavior using visual feedback information extracted from single or multiple cameras, and either using direct or indirect error computation of visual features. Detailed reviews on visual can be found in [5], [6]. In the case of direct method, the visual servoing control law output is computed directly using the extracted visual features from the camera image. This method is also known as an Image Based Visual Servoing (IBVS) method [7], [8]. Typically IBVS scheme defines the reference signal in the image plane. IBVS maps the error vector in the image space to the robot actuation space. Usually, the target image features are extracted from the raw data of the captured image from camera to compress the salient information; thus IBVS scheme is also known as a feature based scheme or 2D visual servoing. One of the problem with IBVS scheme is that it is difficult to estimate the depth. In the case of indirect method, the extracted visual features are transformed using a pose estimation method to have relative pose between the camera and the target. The visual servoing control law ouput is obtained using the pose error in 3D space between the camera and the target, such system is known as a Position Based Visual Servoing (PBVS) [9]. Therefore a PBVS scheme can overcome the IBVS issue of the depth estimation. Recently a detailed comparison of the two basic visual servoing schemes in the context of stability and robustness with respect to system modelling error was presented in [10]. In term of the camera configuration, both basic visual servoing schemes can be applied using eye-in-hand or eye-to-hand configurations. In the eye-in-hand configuration, one or multiple cameras are placed on the robot platform observing the target [11]. In contrast, in the eye-to-hand configuration, one or multiple cameras are placed permanently in such a way, the movement of the robot and the target can be observed [12].

With respect to the mobile robot navigation field of study, many articles have focused on the design of PBVS-like methods [4], [13], [14]. This paper presents the analytical development of an Image Based Visual Servoing method used for the differential drive mobile robot navigation. The developed control law algorithm is applied on a Beaglebone Black embedded system. The rest of the paper is organised as follows. Section 2 discusses the development of the proposed IBVS control algorithm, Section 3 presents the stability analysis followed by the discussion of the IBVS robustness due to camera callibration error in Section 4. Section 5 shows the experimental results and followed by Section 6 for the conclusion.

II. IBVS for a Differential Drive Mobile Robot

A. Differential Drive Mobile Robot Kinematics

The most popular type of indoor mobile robot is a differential drive system. This system uses two main wheels where each of which is connected to its own motor. A third wheel functions to balance the robot structure and rolls passively. To develop a simple model based of the differential drive contraints, only two parameters are necessary to be measured. The first parameter is the distance between the centre of the left wheel and the right wheel, denoted as L. The second parameter is the wheel radius, r. The instantaneous changes

Indrazno Siradjuddin, PhD. Electrical Engineering Department, Malang State Polytechnic, Indonesia, indrazno@polinema.ac.id

Dr. Indah Agustien Siradjuddin, Informatics Engineering Department, Trunojoyo University, Indonesia, indah.agustien@if.trunojoyo.ac.id

Supriatna Adhisuwignjo, MT., Electrical Engineering Department, Malang State Polytechnic, Indonesia, supriatna@polinema.ac.id

of the direction ω_d and the robot position v_d described in a plannar working space are specified by the action vector $\boldsymbol{\omega} = [\omega_r, \omega_l]^{\mathrm{T}}$, the two angular velocities of right and left wheels. There exists a point along common rotation axis of right and left wheel that is considered as a robot rotation centre, which also known as Instantaneous Centre of Curvature (ICC), see Figure 1. It can be easily deduced that the relationship between the robot angular velocity about the ICC at Δt and the instantaneous translational velocity of each wheel can be described as

$$\omega_d \left(R + \frac{L}{2} \right) = v_r \quad (1)$$

$$\omega_d \left(R - \frac{L}{2} \right) = v_l \quad (2)$$

$$\omega_d R = v_d \quad (3)$$

where R is the distance between the robot frame $\mathcal{F}_d$ origin and the ICC. The translational velocity of right and left wheels denoted as v_r and v_l, respectively. The robot translational velocity v_d can be computed as

$$v_d = \frac{v_r + v_l}{2} \quad (4)$$

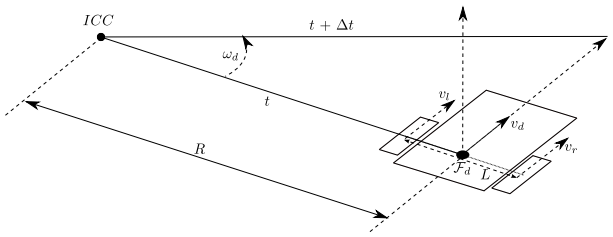

Fig. 1: Instantaneous Centre of Curvature (ICC)

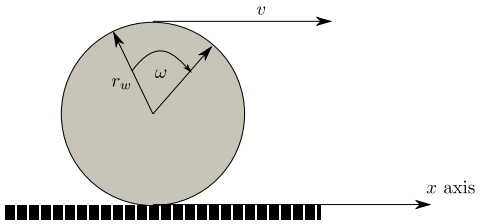

Fig. 2: Rotational and translational velocities of a wheel

Figure 2 shows the graphical relationship between the translational and the rotational velocities of a wheel that defined as $\omega r_w = v$ where r_w is the wheel radius. Therefore, the robot translational velocity in (4) becomes

$$v_d = \frac{\omega_r r_w + \omega_l r_w}{2} \quad (5)$$

where ω_r and ω_l are the rotational or the angular velocities of the right wheel and the left wheel, respectively. From (3) and (4), it can be easily verified that

$$\omega_d = \frac{v_r + v_l}{2R} = \frac{\omega_r r_w + \omega_l r_w}{2R} \quad (6)$$

with little work on (1) and (2), the distance R between ICC and the $\mathcal{F}_d$ origin can be computed using

$$R = \frac{L}{2} \left(\frac{v_l + v_r}{v_r - v_l} \right) \quad (7)$$

Substituting (7) into (8), one can find that

$$\omega_d = \frac{\omega_r r_w - \omega_l r_w}{L} \quad (8)$$

Therefore, equations (5) and (8) can be expressed in a matrix form as

$$\begin{bmatrix} v_d \\ \omega_d \end{bmatrix} = \begin{bmatrix} \frac{r_w}{2} & \frac{r_w}{2} \\ \frac{r_w}{L} & -\frac{r_w}{L} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \quad (9)$$

$$\mathbf{u} = \mathbf{T}\boldsymbol{\omega} \quad (10)$$

Thus (10) maps the action vector $\boldsymbol{\omega}$ in the control space into the action vector $\mathbf{u}$ in the robot working space through a matrix $\mathbf{T}$.

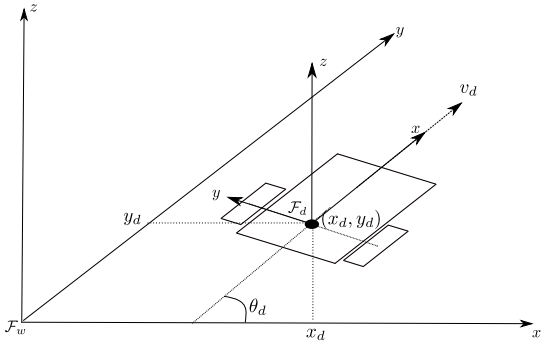

Fig. 3: Mobile robot pose with respect to the world coordinate frame $\mathcal{F}_w$

Now, let $\boldsymbol{\xi}_d = [x_d, y_d, \theta_d]^{\mathrm{T}}$ denotes the pose of the mobile robot with respect to the world frame $\mathcal{F}_w$ as shown in Figure 3. Indeed, it can be concluded that

$$\dot{x}_d = v_d \cos\theta_d \quad (11)$$

$$\dot{y}_d = v_d \sin\theta_d \quad (12)$$

$$\dot{\theta}_d = \omega_d \quad (13)$$

where $\dot{x}_d$ and $\dot{y}_d$ are the instantaneous changes of the mobile robot position in x_d and y_d with respect to the world coordinate frame $\mathcal{F}_w$ affected with the action vector $\mathbf{u}$ and of course that the mobile robot orientation rate $\dot{\theta}_d$ is equivalent with the robot angular velocity ω_d. This can be expressed in the matrix form as follows

$$\begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ 0 \\ 0 \\ 0 \\ \dot{\theta}_d \end{bmatrix} = \begin{bmatrix} \cos\theta_d & 0 \\ \sin\theta_d & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_d \\ \omega_d \end{bmatrix} \quad (14)$$

$$\dot{\boldsymbol{\xi}}_d = \mathbf{H}\mathbf{u} \quad (15)$$

Note that (15) is expressed in a complete 3D motion where translation along z axis, rotation about x and y axes are all zeros. Substituting (15) into (10), the mobile robot velocity kinematic can be obtained as follows

$$\dot{\boldsymbol{\xi}}_d = \mathbf{H}\mathbf{T}\boldsymbol{\omega} \quad (16)$$

$$= \mathbf{J}\boldsymbol{\omega} \quad (17)$$

where $\mathbf{J} \in \mathcal{R}^{6\times 2}$ is called a differential drive mobile robot Jacobian which is computed by multiplying matrix $\mathbf{H}$ with matrix $\mathbf{T}$.

B. Interaction Matrix Development

A general formulation of visual servoing systems is obtained by interpreting the visual servoing problem as a task regulation problem. The task function approach applied to visual servoing systems was introduced in [15]. The task of a wide variety of visual servoing systems is to minimise the error $\mathbf{e}(t)$ between the current and the desired image features. A general representation of the visual servoing task is given by

$$\mathbf{e}(t) = \mathbf{s}(\mathbf{g}(t), \mathbf{a}) - \mathbf{s}^* \in \mathcal{R}^k \tag{18}$$

where a set of the current image features $\mathbf{s}$ is defined as a function of an image measurement vector $\mathbf{g}(t)$ and a vector $\mathbf{a}$ that represents intrinsic camera parameters and additional 3D information associated with the objects. $\mathbf{s}^*$ denotes the desired image features vector and k is the number of image features. The choice of the image features $\mathbf{s}$ is not trivial since one should guarantee that the regulation of $\mathbf{s}(t)$ to the desired $\mathbf{s}^*$ strictly implies regulation of the camera pose $\boldsymbol{\xi}_c$ to the desired camera pose configuration $\boldsymbol{\xi}_c^*$ associated with $\mathbf{s}^*$. Therefore, it also implies that the motion of the image feature $\mathbf{s}(\mathbf{g}(t), \mathbf{a})$ is induced by the camera velocity $\dot{\boldsymbol{\xi}}_c = [\mathbf{v}_c, \boldsymbol{\omega}_c]^{\mathrm{T}}$, where $\mathbf{v}_c = [\dot{x}_c, \dot{y}_c, \dot{z}_c]^{\mathrm{T}}$ and $\boldsymbol{\omega}_c = [\omega_x, \omega_y, \omega_z]^{\mathrm{T}}$ are the camera translational velocity and the camera angular velocity, respectively. A standard visual servoing controller diagram is depicted in Figure 4. The relationship between the image

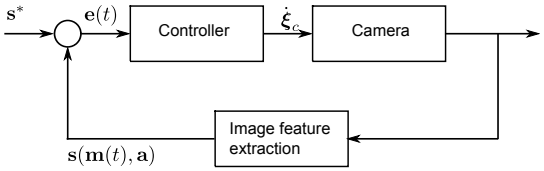

Fig. 4: Basic visual servoing controller diagram

feature velocity $\dot{\mathbf{s}}$ and the camera velocity $\dot{\boldsymbol{\xi}}_c$ is described as

$$\dot{\mathbf{s}} = \mathbf{L}\dot{\boldsymbol{\xi}}_c \tag{19}$$

where $\mathbf{L} \in \mathcal{R}^{k\times m}$ is the Interaction matrix associated with $\mathbf{s}$ and m is the controller DOF; typically $m = k = 6$ to consider full 3D motion.

In this work, an image point is considered as a visual feature. To model a visual servoing system, the projection of an object with respect to the pinhole camera system must be described. Figure 5 illlustrates the camera pinhole model. In Figure 5, the coordinates vector of a 3D point $\mathbf{M} = [x_M, y_M, z_M]^{\mathrm{T}}$ is projected in 2D camera image plane coordinates as $\mathbf{m} = [x_m, y_m]^{\mathrm{T}}$. Note that the image point coordinates vector $\mathbf{m}$ is used as the image features vector $\mathbf{s}$. The projection from 3D into 2D coordinates is described as

$$x_m = \frac{x_M}{z_M} = \frac{u - u_c}{f_c k_u} \tag{20}$$

$$y_m = \frac{y_M}{z_M} = \frac{v - v_c}{f_c k_v} \tag{21}$$

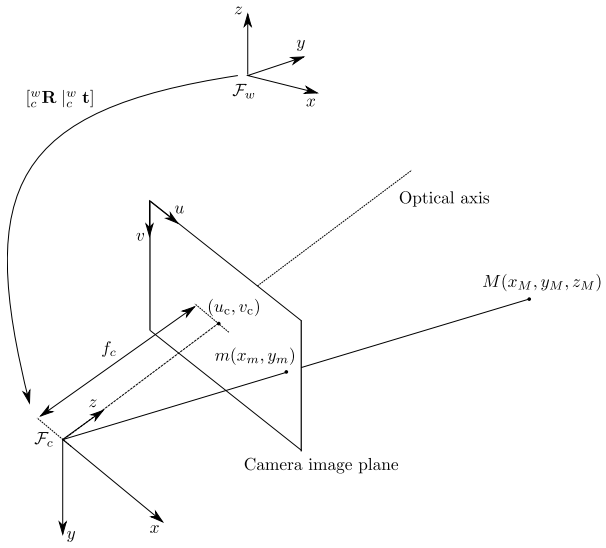

Fig. 5: Camera pinhole model

where (u, v) are the coordinates of the projected point m expressed in pixels, (u_c, v_c) are the image plane principal coordinates and f_c is the camera lens focal length, k_u is the pixel size in the u direction and k_v is the pixel size in the v direction. Lets first, consider the transformation of (x_M, y_M) into (x_m, y_m). The first derivative of (20) and (21) are derived as

$$\dot{x}_m = \frac{\dot{x}_M z_M - x_M \dot{z}_M}{(z_M)^2} \tag{22}$$

$$\dot{y}_m = \frac{\dot{y}_M z_M - y_M \dot{z}_M}{(z_M)^2} \tag{23}$$

The mapping of the camera velocity into the velocity of a 3D point can be described using a well-known formula as follows

$$\dot{\mathbf{M}} = -\mathbf{v}_c - [\boldsymbol{\omega}_c]_\times \mathbf{M} \tag{24}$$

where the skew symmetric matrix of $[\boldsymbol{\omega}_c]_\times$ is computed as

$$[\boldsymbol{\omega}_c]_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{25}$$

Thus we have

$$\begin{aligned} \dot{x}_M &= -\dot{x}_c - \omega_y z_M + \omega_z y_M \\ \dot{y}_M &= -\dot{y}_c - \omega_z x_M + \omega_x z_M \\ \dot{z}_M &= -\dot{z}_c - \omega_x y_M + \omega_y x_M \end{aligned} \tag{26}$$

Substituting (26) into (22) and (23), the velocity of the projected point on the image plane associated with the camera movement described in (19) can be formulated where $\dot{\mathbf{s}} = [\dot{x}_m, \dot{y}_m]^{\mathrm{T}}$ and the interaction matrix $\mathbf{L} \in \mathcal{R}^{2\times 6}$ is derived as

$$\mathbf{L} = \begin{bmatrix} -\frac{1}{z_M} & 0 & \frac{x_m}{z_M} & x_m y_m & -(1+x_m^2) & y_m \\ 0 & -\frac{1}{z_M} & \frac{y_m}{z_M} & (1+y_m^2) & -x_m y_m & -x_m \end{bmatrix} \tag{27}$$

Using a single point as image feature does not fullfill the requirement for controlling a robot in a 3D working space,

the minimum number of 3 points is required. Let's denote a vector of 3 points as $\mathbf{s} = [\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3]^{\mathrm{T}}$, a new interaction matrix can be obtained by stacking each individual interaction matrix of $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$, therefore $\mathbf{L} = [\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3]^{\mathrm{T}} \in \mathcal{R}^{6\times 6}$.

C. IBVS Control Law

Let's assume that the camera frame $\mathcal{F}_c$ and the robot local frame $\mathcal{F}_d$ is coincide, therefore it can be deduced that the camera velocity $\dot{\boldsymbol{\xi}}_c = \dot{\boldsymbol{\xi}}_d$. Substituting (17) into (19) yields

$$\dot{\mathbf{s}} = \mathbf{L}\mathbf{J}\boldsymbol{\omega} \tag{28}$$

thus

$$\boldsymbol{\omega} = \mathbf{J}^{\dagger}\mathbf{L}^{\dagger}\dot{\mathbf{s}} \tag{29}$$

where $\mathbf{J}^{\dagger}$ and $\mathbf{L}^{\dagger}$ are generalised inverse of the robot Jacobian and the interaction matrix, respectively. Practically, the desired image features vector is constant, for instance the centre coordinates of the desired image features are on the centre of the camera image view. Therefore, in a simple form, the first derivative of (18) is expressed as $\dot{\mathbf{e}} = \dot{\mathbf{s}}$. The IBVS control law is designed to exponentially decrease the error $\mathbf{e}$, thus it can be formulated that $\dot{\mathbf{e}} = -\lambda\mathbf{e}$ where λ is a positive constant describing how fast the error is regulated to zero. In the case where the camera frame is coincide with the robot frame, the IBVS control algorithm is computed as follows

$$\boldsymbol{\omega} = -\lambda\mathbf{J}^{\dagger}\mathbf{L}^{\dagger}\mathbf{e} \tag{30}$$

In most common case, the camera frame is not coincide with the robot frame. Thus, there is a transformation between the camera frame and the robot frame which denoted by its translation vector $\mathbf{t}_c^d$ and rotation matrix $\mathbf{R}_c^d$ as shown in Figure 6.

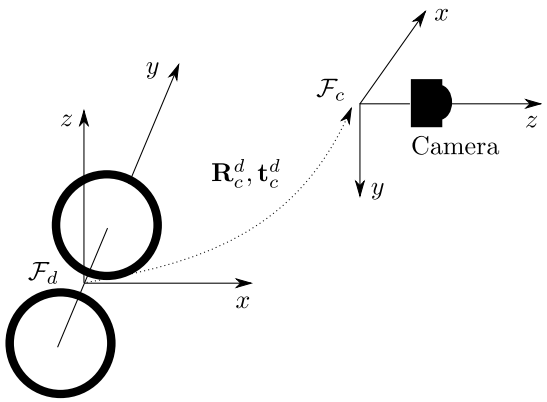

Fig. 6: Transformation between camera frame and robot frame

To relate the 3D velocity vector between coordinate frame, the velocity twist matrix is introduced into control law computation (30) as follows

$$\boldsymbol{\omega} = -\lambda\mathbf{J}^{\dagger}[\mathbf{V}_d^c]^{-1}\mathbf{L}^{\dagger}\mathbf{e} \tag{31}$$

where the velocity twist of the robot frame with respect to the camera frame $\mathbf{V}_d^c \in \mathcal{R}^6 \times 6$ is computed by

$$\mathbf{V}_d^c = \begin{bmatrix} \mathbf{R}_d^c & [\mathbf{t}_d^c]_{\times}\mathbf{R}_d^c \\ \mathbf{0} \in \mathcal{R}^{3\times 3} & \mathbf{R}_d^c \end{bmatrix} \tag{32}$$

and the skew symmetric matrix of the robot frame origin position with respect to camera frame ($\mathbf{t}_d^c = [t_x, t_y, t_z]^{\mathrm{T}}$) is expressed as

$$[\mathbf{d}_d^c]_{\times} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{33}$$

Using basic transformation formula between coordinate systems, the relation of the homogeneous transformation matrix components between the camera frame and the robot frame can be obtained by

$$\mathbf{R}_d^c = [\mathbf{R}_c^d]^{\mathrm{T}} \tag{34}$$

$$\mathbf{t}_d^c = -[\mathbf{R}_c^d]^{\mathrm{T}}\mathbf{t}_c^d \tag{35}$$

Indeed, the computation of the IBVS control law in (30) and (31) require the exact measurement of the robot physical model parameters such as: robot wheel diameter r_w and L. In practice, due to the mechanical imprecission design, it is more ellegant to denote $\widehat{\mathbf{J}}$ instead of $\mathbf{J}$ for representing the estimated robot Jacobian. It is also noted that the interaction matrix $\mathbf{L}$ in (27) is the function of the exact 3D and 2D parameters of the target object: Z_M, x_m and y_m. Those parameters, in fact can only be estimated considering the camera intrinsic parameters (e.g f_c) and extrinsic parameters ($\mathbf{R}_d^c$ and $\mathbf{t}_d^c$) difficult to be obtained precisely during callibration process. Furthermore, a complex algorithm of pose estimation should be used in order to compute the estimated interaction matrix denoted as $\widehat{\mathbf{L}}$ in every iteration of visual servoing process. Thanks to Chaumette and Hutchinson [5] for the detail discussion of the interaction matrix option where the desired target parameters are used. In this option, the computation of the interaction matrix is simpler where the computation of a complex pose estimation algorithm is not required while the exponential decreasing error can still be achieved. Moreover, the estimated interaction matrix $\widehat{\mathbf{L}}$ becomes constant, it is not necessary that $\widehat{\mathbf{L}}$ has to be computed in every iteration. In this work, the estimated robot Jacobian and interaction matrix of the desired target were used, therefore, the expression of the IBVS control law is described as

$$\boldsymbol{\omega} = -\lambda\widehat{\mathbf{J}}^{\dagger}\widehat{\mathbf{L}}^{\dagger}\mathbf{e} \tag{36}$$

or

$$\boldsymbol{\omega} = -\lambda\widehat{\mathbf{J}}^{\dagger}[\widehat{\mathbf{V}}_d^c]^{-1}\widehat{\mathbf{L}}^{\dagger}\mathbf{e} \tag{37}$$

if the camera frame is not coincide with the robot frame.

III. Stability Analysis

The stability of the proposed closed-loop system (36) or (37) can be verified in terms of Lyapunov stability. Lets define the quadratic Lyapunov candidate function as

$$V = \mathbf{e}^{\mathrm{T}}\mathbf{e} \tag{38}$$

Thus, the time derivative of the quadratic Lyapunov candidate function (38) can be computed as

$$\dot{V} = \mathbf{e}^{\mathrm{T}}\dot{\mathbf{e}} \tag{39}$$

Lets denote $(\mathbf{LJ})$ as $\mathbf{A}$, $(\mathbf{J}^{\dagger}\mathbf{L}^{\dagger})$ as $\mathbf{A}^{\dagger}$. The estimated of those two matrices denoted as $\widehat{\mathbf{A}}$ and $\widehat{\mathbf{A}}^{\dagger}$, respectively. Using $\dot{\mathbf{e}} = \dot{\mathbf{s}}$ and equation (29), the time derivative of the quadratic Lyapunov candidate function (39) can be expanded as

$$\dot{V} = -\lambda \mathbf{e}^{\mathrm{T}} \mathbf{A} \boldsymbol{\omega} \tag{40}$$

In fact, the angular velocity vector $\boldsymbol{\omega}$ was computed using the estimated robot jacobian and the estimated interaction matrix, therefore $\dot{V}$ can be obtained by

$$\dot{V} = -\lambda \mathbf{e}^{\mathrm{T}} \mathbf{A} \widehat{\mathbf{A}}^{\dagger} \mathbf{e} \tag{41}$$

$$\dot{V} = -\lambda \mathbf{e}^{\mathrm{T}} \mathbf{A} (\widehat{\mathbf{A}}^{\dagger} + \mathbf{A}^{\dagger} - \mathbf{A}^{\dagger}) \mathbf{e} \tag{42}$$

$$\dot{V} = -\lambda \mathbf{e}^{\mathrm{T}} \mathbf{A} \mathbf{A}^{\dagger} \mathbf{e} - \lambda \mathbf{e}^{\mathrm{T}} \mathbf{A} \boldsymbol{\epsilon} \mathbf{e} \tag{43}$$

where $\boldsymbol{\epsilon}$ is the error between the estimated and the actual of the inverse robot Jacobian and the interaction matrix, $\widehat{\mathbf{A}}^{\dagger}$ and $\mathbf{A}^{\dagger}$, respectively. The system asymptotic stability can be achieved if condition below is ensured:

$$\|\mathbf{A}\mathbf{A}^{\dagger}\| = \|\mathbf{I}\| > \|\mathbf{A}\boldsymbol{\epsilon}\| \tag{44}$$

Indeed, if the estimated robot Jacobian and the interaction matrices are equal to the actual values, $\boldsymbol{\epsilon} = \mathbf{0}$ then the system asymptotic stability is guaranteed, $\dot{V} \leq 0$.

IV. IBVS Robustness Due to Camera Callibration Error

It is also necessary to prove that the IBVS control law is a robust system due to the camera callibration error. Firtly, lets recall the fundamental 3D-2D mapping from a pin-hole camera model which described as follows:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_c k_u & 0 & u_c \\ 0 & f_c k_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_m \\ y_m \\ 1 \end{bmatrix} \tag{45}$$

$$\mathbf{x}_i = \mathbf{W}\mathbf{x}_m \tag{46}$$

By considering that the intrinsic camera parameter matrix $\widehat{\mathbf{W}}$ is used and the fact that the IBVS uses direct image measurement on the image space as the contoller input, it can be deduced that

$$\widehat{\mathbf{x}}_m = \widehat{\mathbf{W}}^{-1}\mathbf{x}_i \tag{47}$$

Substituting (46) into (47)

$$\widehat{\mathbf{x}}_m = \widehat{\mathbf{W}}^{-1}\mathbf{W}\mathbf{x}_m \tag{48}$$

where

$$\widehat{\mathbf{W}}^{-1}\mathbf{W} = \begin{bmatrix} \frac{f_c k_u}{\widehat{f_c}\widehat{k_u}} & 0 & \frac{u_c - \widehat{u}_c}{\widehat{f_c}\widehat{k_u}} \\ 0 & \frac{f_c k_v}{\widehat{f_c}\widehat{k_v}} & \frac{v_c - \widehat{v}_c}{\widehat{f_c}\widehat{k_u}} \\ 0 & 0 & 1 \end{bmatrix} \tag{49}$$

$$= \begin{bmatrix} \widehat{K}_u & 0 & \widehat{U} \\ 0 & \widehat{K}_v & \widehat{V} \\ 0 & 0 & 1 \end{bmatrix} \tag{50}$$

Note that f_c, k_u and k_v are always positive. The first derivative of (48) can be formulated as

$$\dot{\widehat{x}}_m = \widehat{K}_u \dot{x}_m \tag{51}$$

$$\dot{\widehat{y}}_m = \widehat{K}_v \dot{y}_m \tag{52}$$

Thus in compact form, it can be deduced that

$$\widehat{\mathbf{L}} = \widehat{\mathbf{K}}\mathbf{L} \tag{53}$$

where

$$\widehat{\mathbf{K}} = \begin{bmatrix} \widehat{K}_u & 0 \\ 0 & \widehat{K}_v \end{bmatrix} > 0 \tag{54}$$

Lets consider that $\mathbf{J} \approx \widehat{\mathbf{J}}$ then the first derivative of the defined Lyapunov function can be expressed as

$$\dot{V} = -\lambda \mathbf{e}^{\mathrm{T}} \mathbf{L} \widehat{\mathbf{L}}^{\dagger} \mathbf{e} \tag{55}$$

$$= -\lambda \mathbf{e}^{\mathrm{T}} \mathbf{L}\mathbf{L}^{\dagger} \widehat{\mathbf{K}}^{-1} \mathbf{e} \leq 0 \tag{56}$$

thus the asymptotic stability of the system can be ensured. Indeed, if the kinematic robot Jacobian formulation is too far from the actual then the system can be unstable.

V. Experimental Results

A. Experimental Setup

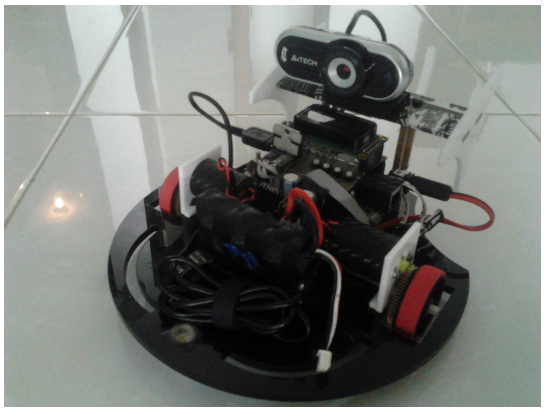

Fig. 7: A differential drive mobile robot prototype

The experimental setup uses a a differential drive mobile robot prototype with a webcam attached on the robot platform as shown in Figure 7. Two small DC motors were used to rotate the robot wheels with belt transmission technique to gain the torque. Two channels encoder system is used on both wheel drives to measure the angular velocity vector of both robot wheels, ω_r and ω_l. A microcontroller board system of an ATMEL ATMEGA 8 was used as an internal robot controller to regulate the robot angular velocity vector $\boldsymbol{\omega}$ at the desired value obtained from the MBVS computation expressed in (31) where its process was run separetly inside the Beaglebone Black embedded system. A simple PD controller method was used to control the angular velocity of the robot wheels, where the PD controller parameters were adjusted using a well known Ziegler-Nichols tuning method. The resulting angular velocity vector command was sent from the Beaglebone Black to the internal controller using UART communication protocol method. The Ubuntu 14.04 kernel was used as an operating system for the Beaglebone Black along with ViSP [16] and OpenCV libraries [17]. The experimental setup used the coordinates of four dots image as image features, therefore, the image feature vector denoted as $\mathbf{s} = [x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4]$

where the subscript 1 to 4 indicate the index of each dot. To simplify the notation for the detail explanation in the next discussion, $\mathbf{s} = [s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8]$ is used instead. Thanks to ViSP libraries that provides functions to extract and track the dot image on the image view of the camera. A simple self-explanatory diagram block of the experimental setup is shown in Figure 8.

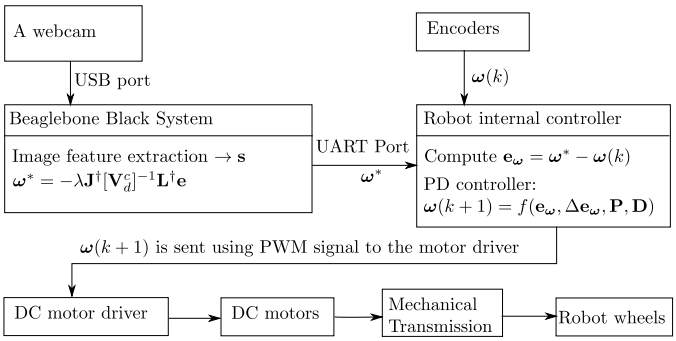

Fig. 8: Diagram block of the experimental setup

B. Simulation

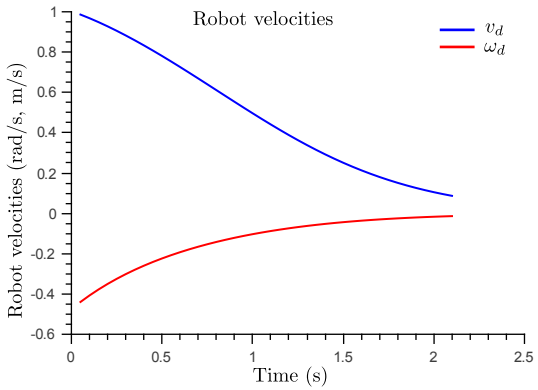

Fig. 9: Robot velocities

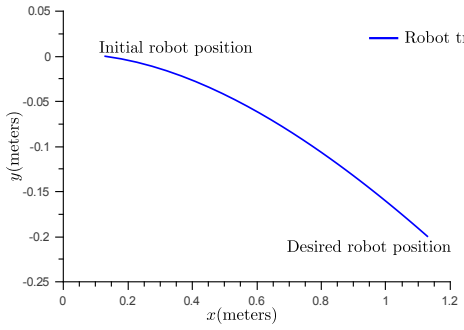

Fig. 10: Robot trajectory

A simulation test was conducted to complement the system analysis. The simulation analysis used the robot velocities, the robot trajectory, the image feature error, the norm of the image feature error and the robot wheel angular velocities

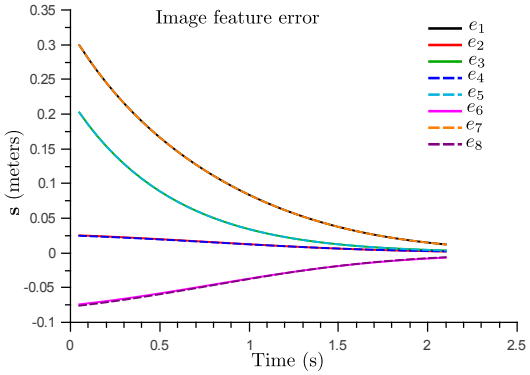

Fig. 11: Image feature error

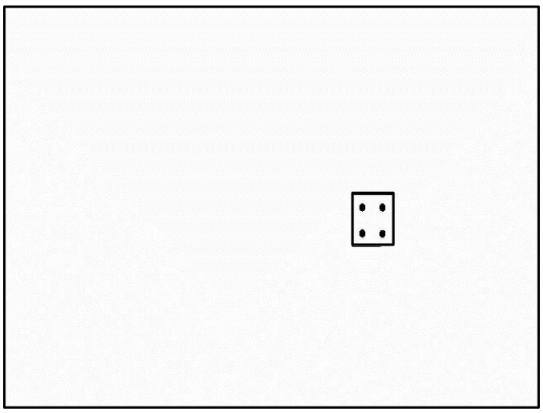

Fig. 12: Image feature error at initial robot pose

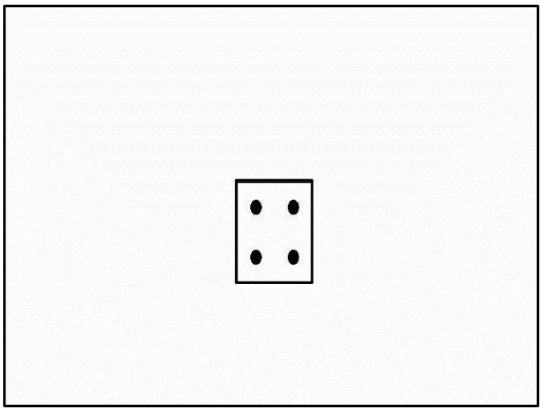

Fig. 13: Image feature error at desired robot pose

as the measured performance variables. Figure 9 shows the robot translational and the angular velocites expressed in robot local frame. It is shown that the robot velocity components were converged to zero. The robot moved with relatively high speed from initial position and stop when it was at a desired position. The robot trajectory in 3D working space is drawn in Figure 10. The corresponding image feature error resulted by the robot movement from initial to the desired position is shown in Figure 11. Note that the measured image feature errors are described in meters instead in pixels. The conversion

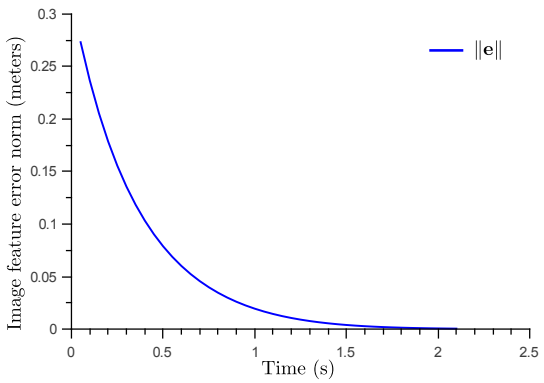

Fig. 14: Norm of the image feature error

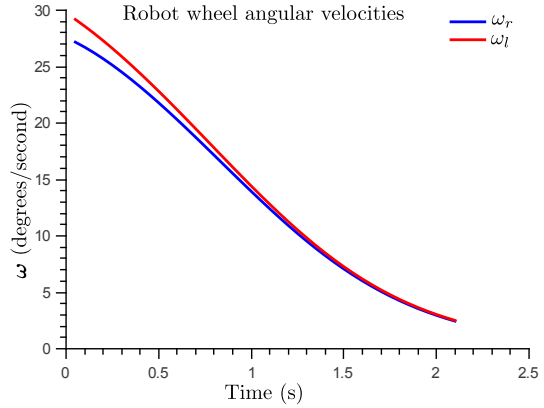

Fig. 15: Robot wheel angular velocities

from pixels to meters were done using a simple pinhole camera model. Within the camera image view, the four dots image at the initial robot pose and the desired robot pose were captured as shown in Figure 12 and Figure 13, respectively. The trajectory of the norm of the image feature error is plotted in Figure 14. The error norm trajectory decreased exponentially as expected. Finally, the smooth convergence of the output of the MBVS control method, $\boldsymbol{\omega}$, was achieved as shown in Figure 15.

C. Realtime

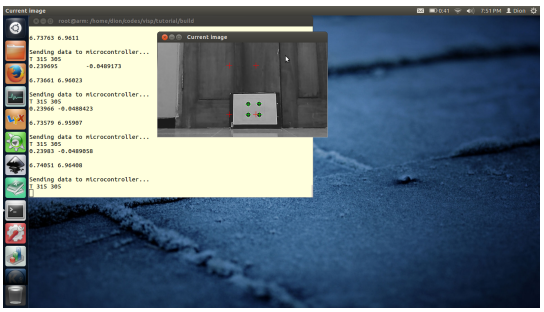

Fig. 16: A captured ubuntu terminal of the Beaglebone Black

Realtime experiments were conducted succesfully. In this section, some images were captured from one of the experiments to illustrate the performance of the MBVS method. Figure 16 shows the captured ubuntu terminal run on the Beaglebone Black during the robot motion. It is shown that the terminal printed the program messages to send the velocity command to the robot internal controller through UART communication port.

Fig. 17: Bird views of the captured robot motion

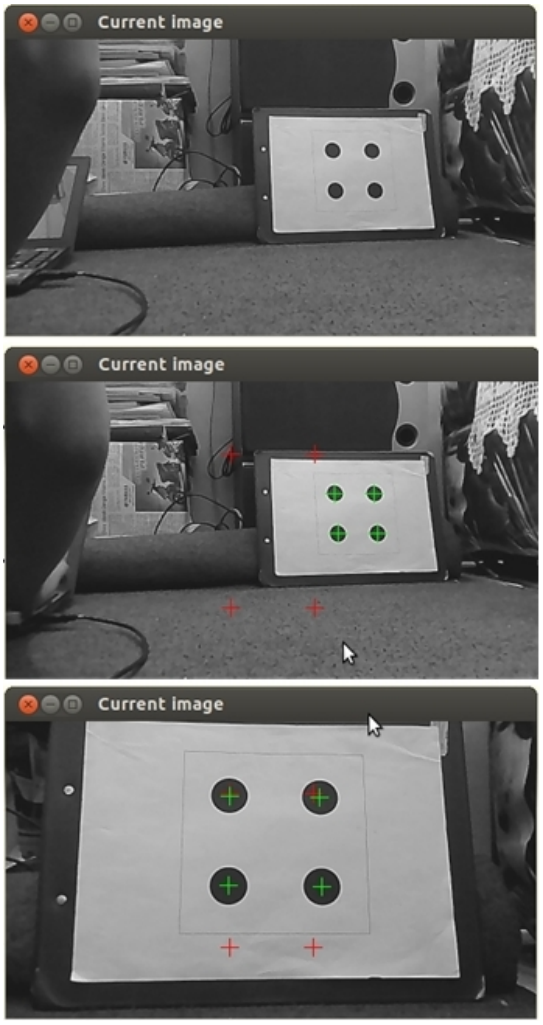

Fig. 18: Captured image feature during the robot motion

A sequence of bird view images were taken to show the realtime experimental setup as well as the motion of the robot from initial position towards the target position (see Figure 17), such that the current image features brought to the desired image features position in thge camera view (see Figure 18).

VI. Conclusion

An IBVS control strategy for a differential drive mobile robot navigation using single camera has been discussed in this paper. The detail fundamental development of the method has been presented. The stability and the system robustness in the presence of the camera callibration error have been discussed in detail. The perfomance analysis has been carried out in both simulation and realtime experiments using a differential drive robot prototype. The coordinates of the four dots image were used as the image features. The trajectory of the image features vector and its errors, the robot pose trajectory, and also the robot velocities have been shown to show the performance of the MBVS system. It has been shown that the norm of the image feature error vector decreased exponentially. In realtime experiments, photos have been taken to show the setup and also the motion of the robot from the initial position to the desired position.

Acknowledgments

The authors would like to thank to the Indonesian Directorate General of the Higher Education (DIKTI) and Malang State Polytechnic who have supported this research project.

References

[1] H. Rashid and A. K. Turuk, "Dead reckoning localization technique for mobile wireless sensor networks." *CoRR*, vol. abs/1504.06797, 2015.

[2] H. Bao and W.-C. Wong, "A novel map-based dead-reckoning algorithm for indoor localization." vol. 3, no. 1, pp. 44–63, 2014.

[3] A. Rudolph, "Quantification and estimation of differential odometry errors in mobile robotics with redundant sensor information." *I. J. Robotic Res.*, vol. 22, no. 2, pp. 117–128, 2003.

[4] H. M. Becerra and C. Sags, "Pose-estimation-based visual servoing for differential-drive robots using the 1d trifocal tensor." in *IROS*. IEEE, 2009, pp. 5942–5947.

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

[6] ——, "Visual servo control. ii. advanced approaches [tutorial]," *Robotics Automation Magazine, IEEE*, vol. 14, no. 1, pp. 109–118, March 2007.

[7] I. Siradjuddin, T. McGinnity, S. Coleman, and L. Behera, "A computationally efficient approach for jacobian approximation of image based visual servoing for joint limit avoidance," in *Mechatronics and Automation (ICMA), 2011 International Conference on*, Aug 2011, pp. 1362–1367.

[8] I. Siradjuddin, L. Behera, T. McGinnity, and S. Coleman, "Image-based visual servoing of a 7-dof robot manipulator using an adaptive distributed fuzzy pd controller," *Mechatronics, IEEE/ASME Transactions on*, vol. 19, no. 2, pp. 512–523, April 2014.

[9] ——, "A position based visual tracking system for a 7 dof robot manipulator using a kinect camera," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, June 2012, pp. 1–7.

[10] F. Janabi-Sharifi, L. Deng, and W. J. Wilson, "Comparison of basic visual servoing methods," *Mechatronics, IEEE/ASME Transactions on*, vol. 16, no. 5, pp. 967–983, Oct 2011.

[11] P. Cigliano, V. Lippiello, F. Ruggiero, and B. Siciliano, "Robotic ball catching with an eye-in-hand single-camera system," *Control Systems Technology, IEEE Transactions on*, vol. 23, no. 5, pp. 1657–1671, Sept 2015.

[12] V. Lippiello, B. Siciliano, and L. Villani, "Position-based visual servoing in industrial multirobot cells using a hybrid camera configuration," *Robotics, IEEE Transactions on*, vol. 23, no. 1, pp. 73–86, Feb 2007.

[13] D. Jung, J. Heinzmann, and A. Zelinsky, "Range and pose estimation for visual servoing of a mobile robot." in *ICRA*. IEEE Computer Society, 1998, pp. 1226–1231.

[14] A. Cherubini, F. Chaumette, and G. Oriolo, "A position-based visual servoing scheme for following paths with nonholonomic mobile robots." in *IROS*. IEEE, 2008, pp. 1648–1654.

[15] P. Rives, F. Chaumette, and B. Espiau, "Visual servoing based on a task function approach," in *Experimental Robotics I*, ser. Lecture Notes in Control and Information Sciences, V. Hayward and O. Khatib, Eds. Springer Berlin Heidelberg, 1990, vol. 139, pp. 412–428. [Online]. Available: http://dx.doi.org/10.1007/BFb0042532

[16] E. Marchand, F. Spindler, and F. Chaumette, "Visp for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robotics and Automation Magazine*, vol. 12, no. 4, pp. 40–52, December 2005.

[17] G. Bradski, *Dr. Dobb's Journal of Software Tools*, 2000.