Proceedings of the 33rd Chinese Control Conference
July 28-30, 2014, Nanjing, China

A Visual Servoing Docking Approach for Marsupial Robotic System

Zhao Peng[1], Cao Zhiqiang[1], Xu Lingyi[1,2], Zhou Chao[1], Xu De[1]

1. Institute of Automation, Chinese Academy of Sciences, Beijing, 100190
E-mail: {peng.zhao, zhiqiang.cao, chao.zhou}@ia.ac.cn

2. University of Science and Technology Beijing, Beijing 100083, China
lingyixu@yahoo.com

Abstract: This paper presents a visual servoing docking approach for marsupial robotic system. A vertical V-shaped visual benchmark is designed to guide the docking motion of a child robot with the feedback of image features. By changing the angular of the camera with a rotational DOF, the child robot adjusts its heading orientation for docking. The task is modeled into six states: Blind, Adistance, Atangent, Around, Aiming, and Parking states. The transform conditions among the states are also given. To verify the approach, a simulation platform is designed, including an image-infor virtual machine, a decision-making unit and a pose refresher. The simulation results verify the presented docking approach.

Key Words: Marsupial Robotic system, docking, visual servoing, image-infor virtual machine.

1 Introduction

The marsupial robotic system is a heterogeneous robotic composition, benefiting from the combination of the mother robot's navigation strength and the child robots' adaptability [1][2][3], which offers an important perspective to improve the performance of robotic systems. Many research efforts are put into this collaboration mode including UGV-UGV cooperation, UGV-UAV alliance, USV-UAV system, UAV-UAV system and even the space aircrafts applications. UGV-UGV type receives more attentions. The MACS-RACS[4] system is designed for indoor surveillance, and the "mother" MACS has better navigation ability with more task sensors, and it will release the small-sized "child" RACS when facing a narrow room. The Silver Bullet-Bujold[5] system is researched for urban search and rescue. The child Bujold can change its shape to adapt to the environment, and it needs power and image processing ability from the Silver Bullet. A similar system is researched for Coal Mine Rescue[6]. In [7], An ATRV-Mini robot and four our Sony Aibo legged robots are united. Andrew and his team have been working on marsupial robotic team for many years with contribution in docking station[8], docking approach[9] and coordination[10].

It is a basic requirement for a child robot to dock into the station mounted on the mother robot when the child robot needs transportation, power recharging or refuge when encountering a temporarily harmful environment. Human tele-operation will be very limited because it usually consumes exorbitant data of video. Docking behavior of artificial agent was originally explored by Arkin and Murphy in [11]. The surrounding area of the mother robot or docking station is divided into four sub-regions: ballistic region, coercive region, approach zone, and dead-reckoning zone. Among these zones, docking in the third one is crucial, and the problem to be solved is how to make sure the child robot gets both exact position and orientation with limited sensing information before it gets too close to the entrance.

Kadioglu *et al.* set color markers on the docking station(an upside open box) [9]. During docking, the child robot uses its camera to capture the markers' image. In [12], a visual docking benchmark consisting of two same-sized colored rectangles is printed at the entrance of docking station.

This paper represents a novel vertical V-shaped benchmark that can significantly increase the angular of the docking zone. A rotational degree of freedom is added to the child robot's camera, which allows the docking be much concise with a few image features. To verify the docking approach, a simulation platform is introduced.

The remainder of the paper is organized as follows. Section 2 presents the docking approach. The simulation platform is depicted in Section 3 and simulation results are described in section 4. Section 5 concludes the paper.

2 Docking Approach

2.1 Benchmark and the camera with a rotational DOF

As depicted in Fig. 1, a vertical V-shaped visual benchmark is set upon the docking station, the three points L, B, R are in the same height and form up a English letter "V". The bottom vertex B is just upon the vertical bisector of the entrance, with the other two vertexes L and R being symmetrical. Each vertex is an intersection of three color blocks with a border line under it and can be easily extracted by image processing.

A CMOS camera is installed on the top of the child robot, and it is located on the shaft of a digital servo that endows the camera with a rotational degree of freedom. During camera rotating, its optical axis keeps being parallel to the ground plane. The camera captures the benchmark to extract its image features such as the horizontal coordinates u_L, u_B, u_R of the three vertexes and the length ${}_{A}l$ of the border line l_B under vertex B.

[*]This work is supported in part by the National Natural Science Foundation of China under Grants 61273352, 61175111, 61105105, 61227804, 60805038.

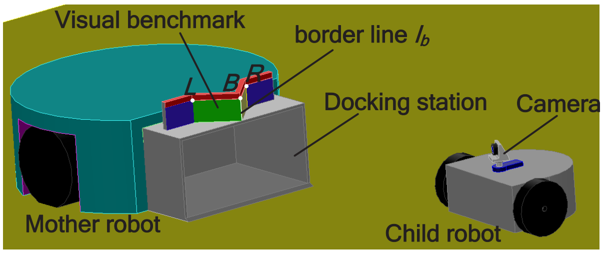

Fig. 1: Visual servoing docking based on the benchmark.

2.2 Docking Description

The docking journey is divided into three stages, as illustrated in the Fig. 2. Firstly, the camera turns to the front of the child robot to help it go ahead to the vertex B to the given distance R_g. Secondly, the servo turns the camera's optical axis orthogonal to robot's heading orientation to keep it navigate along the arc with a radius of R_g around the vertex B until the child robot arrives at the front of the entrance, and then it turns 90° to point at the entrance and then docks into the station directly. In the first stage, the distance R_g is achieved by controlling the navigating linear speed with the feedback of the image feature $_{\Delta}l$. During the whole docking journey, the robot heading is controlled by keeping the vertex B near the bisector line of image view. Obviously, it is required that the camera finds the benchmark before the docking journey starts.

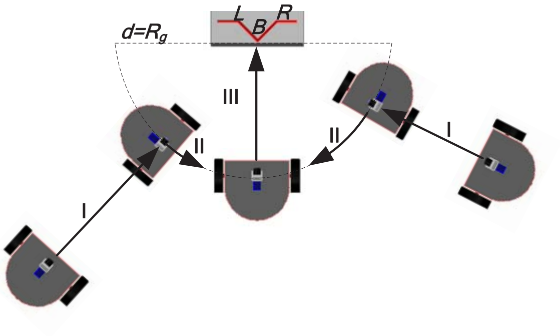

Fig. 2: Motion stages

Specially, the second stage is the core of the docking approach. At the end of first stage, at least two of the vertexes are captured with a proper R_g. The angle of digital servo adjusts gradually to 90° or -90° (90° corresponding to the left side of the entrance), while the robot body rotates to keep the camera pointing at the vertex B. Then the heading orientation is tangent to the arc and the robot starts to run along the arc with a fixed linear speed v_{set} and an adjusting angular speed w. The initial value of w is v_{set}/R_g if the child robot starts from left side; otherwise, it is initialized as $-v_{set}/R_g$. With the camera optical axis pointing at the Vertex B and orthogonal to the robot's body, the problems still to be resolved are how to infer with which side the arc begins at and when to abort the arc. As the top views of the benchmark shown in Fig. 3, the approach zone in front of the Benchmark is divided into 3 sub-areas S_1, S_2 and S_3. In S_1, the camera cannot capture vertex L, whereas in S_2 the vertex R could not be picked.

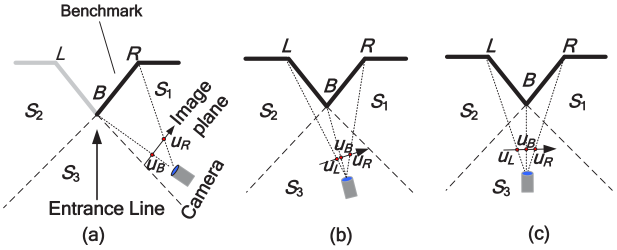

Fig. 3: Top views of the V-shaped benchmark and the camera

In S_3, each of the three vertexes could be captured. It is concluded that the segment length between vertex R and vertex B is longer than that between the vertex L and vertex B if the camera is at the right of the entrance line. There will be an opposite result at the other side. Then which side the camera is at can be judged by formula (1). It is noted that u_R=0 if vertex R is not captured and u_L is endowed with the image width if vertex L is not captured.

$$V_{side}=\begin{cases}0 & u_L+u_R-2u_B\approx 0\\ \mathrm{sgn}(u_L+u_R-2u_B) & else\end{cases} \tag{1}$$

where the value of V_{side} is positive when the camera is at the right side, negative if it is at the other side. When V_{side} is zero, the child robot just arrives at the front of entrance and it should stop to turn its head orientation pointing at the entrance, which means the arc motion is finished.

2.3 Task Model

Based on the docking description above mentioned, the task is modeled into six states, and the states transform diagram is shown in Fig. 4.

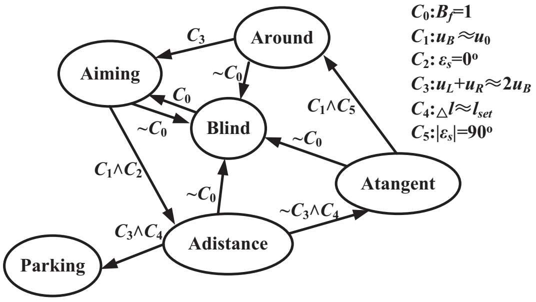

Fig. 4: States transform diagram

Initially, the child robot is in Blind state (B_f=0) and it self-rotates until vertex B has entered into the view of the camera. It turns into Aiming state. During the Aiming state, both the camera servo and the body rotate to make the camera point at vertex B (C_1) and the camera optical axis be identical with the heading orientation (C_2). Then the robot adjusts its distance to the benchmark and it is in state Adistance until C_4 ($_{\Delta}l \approx l_{set}$) is satisfied, where l_{set} is the calibrated $_{\Delta}l$ with the distance to the benchmark being equal to R_g. In this case, if the child robot is just in the front of the entrance (C_3), it directly turns into Parking state, or else, it turns into Atangent state to prepare for arc motion. After its heading orientation is orthogonal to the line between the child robot and the vertex B (C_1C_5), it is in Around state and the child robot starts its arc motion. Next, the child robot turns into Aiming state when C_3 is satisfied.

According to the current task state, the linear velocity v_m and angular velocity w_m of the robot body as well as the rotational angular ε_{sm} of the servo are generated with Algorithm 1, where w_{Self} is self-rotational angular velocity, $f^{w}_{PID}()$ is the angular velocity PID function with the error value of u_0-u_B, $f^{v}_{PID}()$ is the linear velocity PID function, and $S_{AJ}(\varepsilon_t, u_B)$ is used to turn the servo into the target angular ε_t gradually with the vertex B being visible.

Algorithm 1. Decision-making in Different States.

```
input: current task state $m\_state$, the horizontal coordinates
        $u_L$, $u_B$, and $u_R$, the length $_{\Delta}l$.
output: $\varepsilon_{sm}$, $v_m$, $w_m$.
switch ($m\_state$)
    case BLIND:
        $(v_m, w_m, \varepsilon_{sm}) \leftarrow (0, w_{Self}, \varepsilon_{ls})$;
    break;
    case AIMING:
        $(v_m, w_m, \varepsilon_{sm}) \leftarrow (0, f^{w}_{PID}(u_0-u_B), S_{AJ}(0, u_B))$;
    break;
    case ADISTANCE:
        $(v_m, w_m, \varepsilon_{sm}) \leftarrow (f^{v}_{PID}(\triangle l_{set}-{}_{\Delta}l), f^{w}_{PID}(u_0-u_B), 0)$;
        $V_{side} \leftarrow sgn(u_L+u_0-2u_B)$;
    break;
    case ATANGENT:
        $(v_m, w_m, \varepsilon_{sm}) \leftarrow (0, f^{w}_{PID}(u_0-u_B), S_{AJ}(-90V_{side,} u_B))$;
    break;
    case AROUND:
        $(v_m, w_m, \varepsilon_{sm}) \leftarrow (v_{set}, f^{w}_{PID}(u_0-u_B)-V_{side}v_{set}/R_g, -90V_{side})$;
    break;
    case PARKING:
        $(v_m, w_m, \varepsilon_{sm}) \leftarrow (v_{set}, f^{w}_{PID}(u_0-u_B)B_f, 0)$;
    break;
    $\varepsilon_{ls}=\varepsilon_{sm}$;
    end
```

3 Simulation Platform

To verify the proposed approach, a simulation platform is developed, which includes an image-infor virtual machine, a decision-making unit and a pose refresher, as shown in Fig. 5. The image-infor virtual machine calculates image features based on virtual internal parameters of a camera and the pose data from pose refresher, and then these features are sent to the decision-making unit. The outputted motion instructions are used to update the pose with pose refresher modeling the virtual child robot as a one-order inertial system.

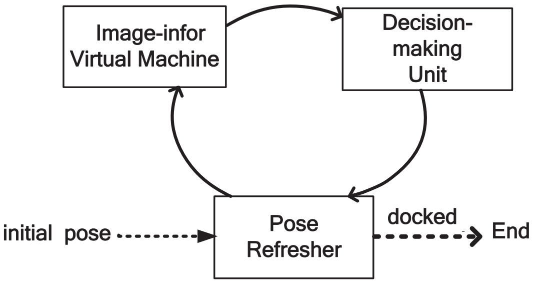

Fig. 5: Simulation platform

3.1 Image-infor Virtual Machine

Fig. 6 shows coordinate frames in the virtual space. The station and the visual benchmark keeps stationary during the docking process and the ground coordinate frame {**G**} is originated at the projection of the vertex B on the ground. X_g axis overlaps on entrance plane's projection line on the ground, whereas Y_g axis is parallel with the normal line of the entrance plane. The pose data sent to image-infor virtual machine includes the child robot's pose $P_g(x_g, y_g, \varepsilon_g)^T$ in {G} and ε_s which is the rotation angular of the camera's digital servo, where ε_g is the angle between X_g axis and child robot's heading. Image-infor virtual machine translates these pose data into transformation matrixes among the world frame {**W**}, robot frame {**R**} and the camera Frame {**C**} in Cartesian space.

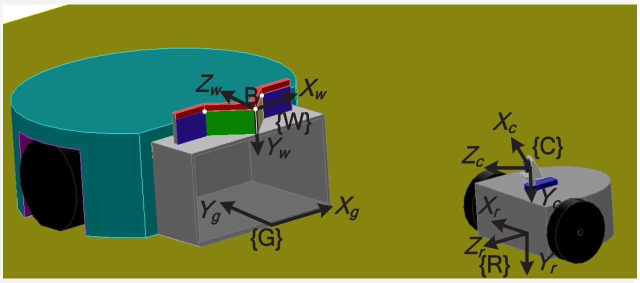

Fig. 6: Coordinate frames in the virtual space

As shown in Fig. 6, the origin of world frame {**W**} is set at vertex B, and its X_w axis is parallel with X_g axis of the frame {**G**}, Y_w axis is vertical to the ground and Z_w axis is parallel with Y_g axis. The robot frame {**R**} is set on the bottom of the child robot with Z_r axis coinciding with its heading. The transformation matrix between {**W**} and {**R**} is provided in formula (2), where h_v is the height of vertex B.

$$^{W}T_{R}=\begin{bmatrix} Rot(Y,\pi/2-\varepsilon_g) & ^{W}P_{R} \\ 0 & 1 \end{bmatrix} \tag{2}$$

$$^{W}P_{R}=\left[x_g, h_v, y_g\right]$$

The camera is fixed with the height of h_c upon the child robot, and the transformation matrix $^{R}T_{C}$ between {**R**} and {**C**} is determined by the rotational angular ε_s of the camera digital motor.

$$^{R}T_{c}=\begin{bmatrix} & 0 \\ Rot(Y,-\varepsilon_s) & -h_c \\ & 0 \\ 0 & 1 \end{bmatrix} \tag{3}$$

The transformation matrix $^{C}T_{W}$ from {**W**} to {**C**} is obtained.

$$^{C}T_{W}=(^{W}T_{R}\,^{R}T_{C})^{-1} \tag{4}$$

Thus, the coordinates of the three vertexes on the image plane can be calculated as follows.

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}=\frac{1}{^{c}z_i}\begin{bmatrix} k_x & 0 & u_0 & 0 \\ 0 & k_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {^{C}T_{W}}\,{^{w}P_i} \tag{5}$$

where $^{w}P_i$ is the coordinates of a vertex in {**W**}. One vertex could be captured only when the formula (6) is satisfied.

$$\begin{cases} {}^{C}z_i > 0 \\ 0 \le u_i \le W \\ 0 \le v_i \le H \end{cases} \tag{6}$$

where W and H are the width and height of the camera view, respectively. As the vertex B is captured, the feature length ${}_{\Delta}l$ of the border line l_B under vertex B is calculated.

$${}_{\Delta}l = k_y l_B / z_c \tag{7}$$

Finally, image-infor virtual machine outputs the state flag B_f to indicate whether vertex B is captured, and the coordinates u_L, u_B, u_R and the length ${}_{\Delta}l$ if B_f is 1.

3.2 Pose refresher

The pose refresher refreshes the current pose data $P_g=(x_g, y_g, \varepsilon_g)^T$ and ε_s of camera's servo to the new one $P_g^*=(x_g^*, y_g^*, \varepsilon_g^*)^T$ and ε_s^* according to the outputs from decision-making unit. Define ${}_{\Delta}T$ is the cycle time imitating the average time consumption of image processing and decision-making for every image.

The child robot is modeled as a one-order inertial system. During every simulating cycle, the immediate linear velocity v(t), angular velocity w(t) and heading angle ε_g(t) are calculated with formula (8).

$$\begin{cases} t \in (0, {}_{\Delta}T) \\ v(t) = v_m + (v_0 - v_m)e^{-t/T_v} \\ w(t) = w_m + (w_0 - w_m)e^{-t/T_w} \\ \varepsilon_g(t) = \varepsilon_{g0} + \int_0^t w(t)d\,t \end{cases} \tag{8}$$

where T_v and T_w are the inertia time constants of the linear and angular velocities, respectively. P_g^* is given by

$$\begin{cases} x_g^* = x_g + \int_0^{\Delta T} v(t)\cos(\varepsilon_g(t))d\,t \\ y_g^* = y_g + \int_0^{\Delta T} v(t)\sin(\varepsilon_g(t))d\,t \,. \\ \varepsilon_g^* = \varepsilon_g(t\,|_{t=\Delta T}) \end{cases} \tag{9}$$

ε_s^* is as follows.

$$\varepsilon_s^* = \varepsilon_{sm} \tag{10}$$

4 Simulation Experiments

In the following simulation experiments, the virtual camera and the parameters of the benchmark refer to Table 1, and the parameters of decision-making are shown in Table 2.

Table 1: Virtual camera and the parameters of the benchmark

Item	Value
image size: L×W (pixel)	640×480
(u_0,v_0)	(320, 240)
(k_x, k_y)	(692.8, 692.8)
world coordinates of vertex B (cm)	(0, 0, 0)
world Coordinates of vertex L (cm)	(-5, 0, 5)
world Coordinates of vertex R (cm)	(5, 0, 5)
Height of the vertexes h_f (cm)	15
Height of the Camera h_c (cm)	8
Length of the border line l_B (cm)	5

Table 2: Internal parameters of decision-making

Item	Value
$\boldsymbol{w_{self}}$	1.0 rad/sec
$\boldsymbol{R_g}$	40cm
$\boldsymbol{l_{set}}$	86 pixel
$\boldsymbol{v_{set}}$	5cm/s
PID coefficients in $\boldsymbol{f^w{}_{PID}}$	(0.005, 0.01, 0.0001)
PID coefficients in $\boldsymbol{f^v{}_{PID}}$	(2, 0.1, 0.05)

The cycle time ${}_{\Delta}T$ is 0.1s, the inertia time constants of the linear and angular velocities are set as T_w=T_v=0.1s.

4.1 Simulation I: Docking without interference

Fig. 7 shows the docking trajectory of the child robot starting from initial location Sc to the location Ec of docking station, and the initial pose P_g is (300.00, -230.00, -π/2). The result verifies the designed docking strategy. At the beginning, the child robot heads toward its park location. When it gets into the given distance, it moves along an arc until arriving in the front of Ec. Finally the child robot parks into the location of the station. The variation of the states is illustrated in Fig. 8, where ***state***=0,1,…,5 is corresponding to Blind, Aiming, Adistance, Atangent, Around and Parking state, respectively. Fig. 9 depicts the variation of the image features u_B and ${}_{\Delta}l$. As illustrated in Fig. 8 and Fig. 9, during the first 3.7s, the child robot is self-rotating in Blind state and no image feature is captured, then it tries to make the u_B to be equal to 320 in Aiming state. After that, it goes into Adistance state and the value of ${}_{\Delta}l$ is increased to be near 86. Then the child robot adjusts its heading orientation in Atangent state, which causes u_B to change apparently. Next, the child robot runs in Around state while the distance from benchmark to the camera is steady and the value of ${}_{\Delta}l$ keeps stable. At the end, it turns into Aiming state for a while and turns into the Parking state until the docking behavior is finished.

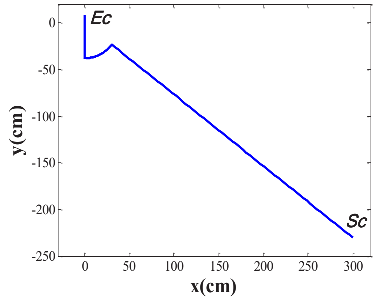

Fig. 7: Docking trajectory of Simulation I

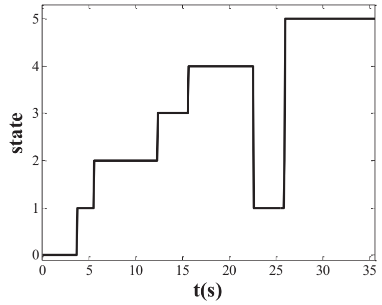

Fig. 8: The variation of states in Simulation I

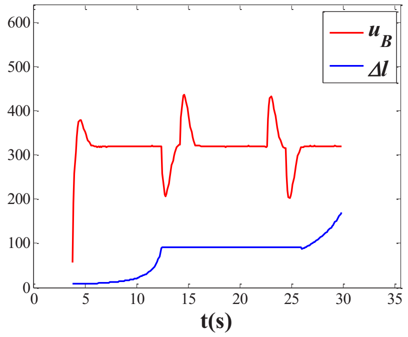

Fig. 9: Variation of the image features u_B and $_{\Delta}l$ of Simulation I

4.2 Simulation II: Docking with interference

To verify the anti-interrupt ability of the given approach, two interferences are imposed during Simulation II. As shown in Fig. 10, the docking behavior starts from location *Sc*, when it arrived at location *A* (t=9.4s), a rotation interference is added and the child robot's heading orientation is clockwise rotated while it still can capture the benchmark. It is still in Adistance state (see Fig. 11). As shown in Fig. 12, u_B floats suddenly and then it resumes to 320, which implies that the interference of rotation is overcome. At the time t=21.9s while the robot is running at Around state, it is dragged from location *B* to location *C* and no image features are captured. Next, the child robot self-rotates and continues to execute the docking smoothly. From the simulations we have conducted, the validity of the proposed approach is verified.

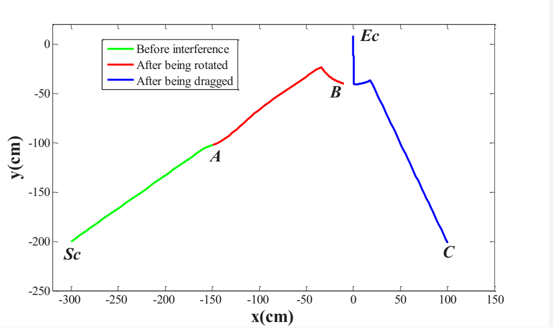

Fig. 10: Docking trajectory of Simulation II

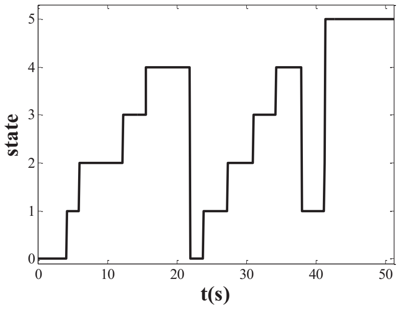

Fig. 11: The variation of states in Simulation II

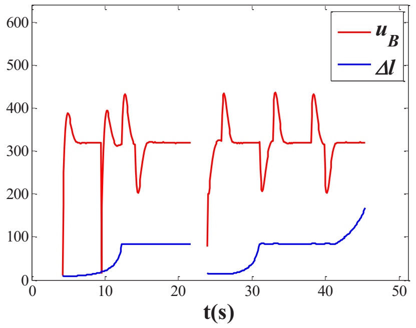

Fig. 12: Variation of the image features u_B and $_{\Delta}l$ of Simulation II

5 Conclusion

A docking approach for the marsupial robotic system based on visual servoing is presented in this paper. The camera of the child robot is added with a rotational DOF. Based on the rotational DOF and the image features of a V-shaped visual benchmark, the child robot adjust its navigation direction to point at docking station or be tangent to an arc, until the child robot arrives at a proper location and pose for parking. The results obtained in simulation platform confirm the validity of the docking approach and its anti-interrupt ability.

The future work will focus on the construction of a marsupial robotic system and the proposed visual servoing docking approach will be applied for the retrieving of child robots.

References

[1] M. Lindemuth, R. Murphy, and E. Steimle, Sea Robot Assisted Inspection, *IEEE Robotics & Automation Magazine*, 18(2): 96-107, 2011.

[2] R. R. Murphy, M. Ausmus, Marsupial-like mobile robot societies, in *Proceedings of 3th International Conference on Autonomous Agents.* 1999: 364-365.

[3] H. Hourani, P. Wolters, A Marsupial Relationship in Robotics: A Survey, in *Intelligent Robotics and Applications*, Springer Berlin Heidelberg, 2011: 335-345.

[4] M. O. Anderson, M. D. McKay, B. S. Richardson, Multirobot Automated Indoor Floor Characterization Team, in *Proceedings of International IEEE International Conference on Robotics and Automation*, 1996: 1750-1753.

[5] R. R. Murphy, Marsupial and shape-shifting robots for urban search and rescue, *IEEE Intelligent Systems and Their Applications*, 15(2): 14-19, 2000.

[6] Jie Zhao, Gang Liu, Yubin Liu, Research on the application of a marsupial robot for coal mine rescue, in *Intelligent Robotics and Applications,* Springer Berlin Heidelberg, 2008: 1127-1136.

[7] Dellaert, Frank, et al, The Georgia Tech Yellow Jackets: A Marsupial Team for Urban Search and Rescue, in *AAAI Mobile Robot Competition,* 2002: 44-49.

[8] C. Carlson, A. Drenner, I. Burt, Modular mobile docking station design, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006: 4722-4727.

[9] E. Kadioglu, N. Papanikolopoulos, A method for transporting a team of miniature robots, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003: 2297-2302.

[10] A. Drenner, M. Janssen, A. Kottas, etc, Coordination and Longevity in Multi-Robot Teams Involving Miniature Robots, *Journal of Intelligent and Robotic Systems*, Vol. 72, Issue 2, pp. 263-284, 2013.

[11] R. C. Arkin, R. R. Murphy, Autonomous navigation in a manufacturing environment, *IEEE Transactions on Robotics and Automation*, 6(4): 445-454, 1990.

[12] B. W. Minten, R. R. Murphy, Low-order-complexity vision-based docking. *IEEE Transactions on Robotics and Automation,* 17(6): 922-930, 2001.