Vision servoing based object following and obstacle avoidance using the Baxter robot

Yuan Zhou
Boston University College of Engineering
110 Cummington Mall, Boston, MA
zhouyuan@bu.edu

Wen Zhou
Boston University College of Engineering
110 Cummington Mall, Boston, MA
zw007981@bu.edu

Abstract

In this paper, we describe a visual servoing control method enhanced by an obstacle avoidance strategy using potential based function. The solution require both relative bearing measurements and relative distance measurements, which acquire from the RGB-camera by color-based detection method through OpenCV. This property allows us to reduce the dependence on the relative distance measurements, which are relatively more inaccurate compared with relative bearing measurements in most cases. Eye-in-hand sensor arrangement can provide a relative high accuracy and avoid vision collision. We also validate our approach by implementation on the Baxter robot.

1. Introduction

Recently, there has been an increasing interest in visual servoing(VS). VS is defined as the use of visual feedback mechanisms for the kinematic control of a robot. Based on the positioning of the camera on the link and control techniques, VS ramifies into several types. Eye-in-hand and eye-to-hand VS are represented by the position of the camera on the robotic manipulator. Being attached on the robot arm, eye-in-hand VS provides a narrower field of view as compared to eye-to-hand servoing.

And the visual servoing control techniques are broadly classified into the follow types [7, 3]: Image-based (IBVS). The control law is based on the error between current and desired features on the image plane, and does not involve any estimate of the pose of the target. The features may be the coordinates of visual features, lines or moments of regions. IBVS has difficulties[2] with motions very large rotations, which has come to be called camera retreat [4]. PBVS is a model-based technique (with a single camera). This is because the pose of the object of interest is estimated with respect to the camera and then a command is issued to the robot controller, which in turn controls the robot. In this case the image features are extracted as well, but are additionally used to estimate 3D information (pose of the object in Cartesian space), hence it is servoing in 3D. In our final project we use PBVS. As for hybrid approaches, they use some combination of the 2D and 3D servoing, like 2-1/2-D Servoing [9], Motion partition-based and Partitioned DOF Based [4].

In visual servoing, we can divide the algorithms into two parts: the first apart is object position based on visual inspection, known the surrounding environment is the first step and this issue can be separated into two parts: vision-based sensor arrangement choice and detection method choice, in which eye-to-hand arrangement and eye-in-hand arrangement are two main classifies. For the first one, the sensor is arranged outside the working space and sense the whole working space, which usually three-party sensor is being used like Kinect sensor. Several relative works have done based on this kind of sensor arrangement. In [22] Xinyu et arrange Kinect sensor on the front upper side of the Baxter robot to get the 3D information of whole robot front view and surrounding environment to do further object self-identification. The camera in [16] is set on the top of the robot to get the 2D top view of the industrial robot arm and a potential function of the robot contour is gotten through the image. For [6] images taken from several stationary cameras in the work cell is presented and general a three views like image system to represent the robot arm and surrounding environment. Eye-to-hand arrangement can provide a large enough view angle and suitable for whole robot though there will have some issues of accuracy due to the resolution of the sensor. For eye-in-hand arrangement, the sensors are mounted on the end-effector of the robot. Specified in Baxter robot, embedding RGB-camera in the robot arm can be used. Sten [17] uses Baxter arm camera to tracking object in horizontal version. In [20] a Kinect camera is installed on the six-legged robot which is similar as eye in hand sensor arrangement. The advantages of eye-in-hand arrangement are providing higher resolution information of the object and obstacles and coordinate sys-

tems are much easier to translate between each other. Eye-in-hand method is chosen in this paper because we focus on the Baxter robot arm object following and obstacles avoid and the whole robot work space is not we are really looking for but the accuracy of the object detection is really important. For detection method choice, most of the relative works using color-based detected method to segment the robot or object from the environment [16, 6, 20]. The reliability of color-based detected method is highly depended on the surrounding environment, object to be detected and sensor property. Particular image processing method needs to used depend on parameters presented above. In [17] the author does not choose detect object directly but use attached ARtag on the object to get the relative position of the tag with respect to the camera and finish the tracking movement. However, ARtag tracking method is basically tracking the ARtag not the object and the object needs to have a flat surface to place the tag. In this paper, color-based detected method is selected because the goal of the object is to guide the robot arm to move so color-based method is enough when we have the right to choose target object.

Another part is about robot motion generation. The goal of formation control is to move a group of agents in order to achieve and maintain a set of desired relative positions. This task has applications in many elds such as surveillance, exploration, and transportation [1, 8, 10, 12, 15, 21]. Formations also allow the control of a large number of agents by a single human operator, and provide robustness to the failure of single agents. In the last decade, people pay more attention to vision-based solutions, where each agent is equipped with an onboard camera (early examples are [5, 11, 14]). In this setting, relative direction measurements (i.e., bearing measurements) are often more reliable than the corresponding distance measurements. Therefore, there has been a recent emphasis on minimizing the use of distance measurements. Among them, Tron [19] et present a formation control solution that can work with bearing measurements alone, or can be augmented with corresponding distances. As we know distance measurements are relatively inaccurate, and by using this robot motion generation method, we can improve the robot's performance for reducing the dependence on the distance measurements.

2. Methodology

2.1. Object position based on visual inspection

Tracking problem formulation In this paper, the tracking object we chose is a tennis ball which has a radius of 0.03 meter. The reason is that the object color is easy to separate from the surrounding captured environment and has a circle contour. For the Baxter robot left arm, its end effector position which is also the position of the embedding left-hand camera is set to a fixed orientation which is vertical to the ground. The position of the end effector can move relate to the detected image information and followed control strategies.

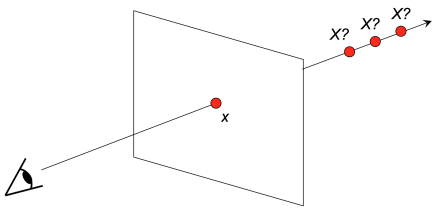

Figure 1. Single-view ambiguity

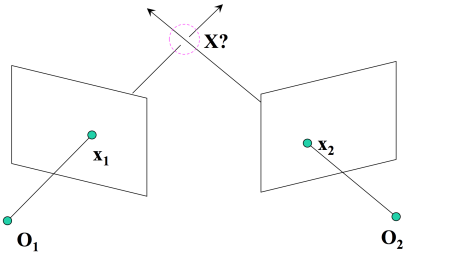

Figure 2. multi-view geometry

Object position based on visual inspection All following tracking algorithm is based on the information which be gotten from the camera image. The basic idea is that we want to recover the coordinate information of 3-D object from the 2-D camera image. Due to the fact that we only use the RGB-camera to detect the object, we can not recovery all the coordinates information of the 3D object using one camera because of Single-view ambiguity showed in Figure 1. We can fix this problem by either known part of the 3-D object information in advance or use multi-view geometry showed in Figure 2.

In this paper, we choose the first method which is getting one coordinate information of the object in advance. There are two way to get the result. First, Baxter robot provides Infrared (IR) Range Sensors on the each arm which can detect the distance between the end pose and the object showed in Figure 3. However, the sensor's performance for relative small object like tennis is poor. The other way is to lock the tennis on a table which means we can get the Z-coordinate of the object in advance, but this solution will let our work environment from 3-D to 2-D. We choose the second method.

Since the tennis will be located on a predefined plane and the camera version is vertical to the table, the arm constraint simplifies the conversion from image pixel coordi-

Figure 3. Baxter hand camera and IR sensor(right)

nates to the robot workspace coordinates. The following plane transformation can be used

$$\begin{bmatrix} P_x \\ P_y \end{bmatrix} = C_c * d * \begin{bmatrix} P_{px} - O_{px} \\ P_{py} - O_{py} \end{bmatrix} + \begin{bmatrix} P_{cx} \\ P_{cy} \end{bmatrix} \quad (1)$$

in which $\begin{bmatrix} P_x \\ P_y \end{bmatrix}$ is the object coordinates based on the Baxter robot which we are looking forward. C_c is the camera calibration factor which is the width of a pixel at one meter. We set it as 0.0025 referenced from [13]. d is the given distance between the camera and table. $\begin{bmatrix} P_{px} \\ P_{py} \end{bmatrix}$ is the detected image pixel coordinates based on the camera frame and we can get it through following OpenCV method. $\begin{bmatrix} O_{px} \\ O_{py} \end{bmatrix}$ is the center pixel coordinates and in this paper, we set the camera resolution as 960x600 and refresh rate is 23.8 fps. $\begin{bmatrix} P_{cx} \\ P_{cy} \end{bmatrix}$ is the camera coordinates based on the Baxter robot and can get from the Baxter built-in function 'get-end-pose'.

2.2. Robot motion generation

General notation In this section, we formula the problem of motion control of the Baxter arm, like what Tron [19] did. We identify the set of N agents as $V = \{1, \ldots, N\}$, and their location as $\{x_i\}_{i \in V}$. We define the distance between nodes $i, j \in V$ as

$$d_{ij} = ||x_j - x_i|| \quad (2)$$

and the bearing direction as

$$\beta_{ij}(x_i, x_j) = d_{ij}^{-1}(x_j - x_i) \quad (3)$$

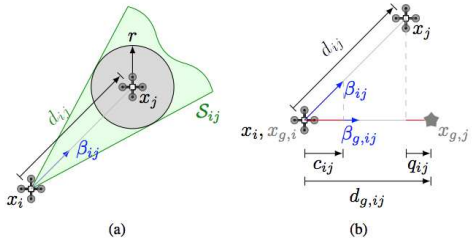

Figure 4. (a) Measurements of agent i with respect to agnet j: bearing β_{ij}, distance d_{ij} and cone S_{ij} for collision avoidance with radius r; (b) The similarity measures c_{ij} Eq. (7), q_{ij} Eq. (8) used in φ, and defined with respect to the desired $\beta_{g,ij}, d_{g,ij}$.

We define a bearing plus distance formation $(\mathcal{F}, \vec{x})$ in which $\vec{x} = stack(\{x_i\}_{i \in V})$ is the configuration of the formation and specifies the position of each agent, and $\mathcal{F} = (V, E_b, E_d)$ is a double graph in which $E_b \in V \times V$ contains the set of pairs (i, j) for which agent i can measure the bearing β_{ij}. We assume each agent i follows the model $\dot{x}_i(t) = u_i$, in which u_i is a control input. So given desired measurements d_g, β_g, our goal in robot motion generation is to design inputs u that drives the agents into a configuration equivalent to x_g. Our control law is the negative gradient of a cost function $\varphi(x)$, i.e. $u = -grad\varphi(x)$.

The cost function The cost function we propose is of the following form:

$$\varphi(x) = \alpha_b \sum_{(i,j) \in E_b} \varphi_{ij}^b(x_i, x_j) + \alpha_d \sum_{(i,j) \in E_d} \varphi_{ij}^d(x_i, x_j) \quad (4)$$

$$\varphi_{ij}^b(x_i, x_j) = d_{ij} f_b(c_{ij}) \quad (5)$$

$$\varphi_{ij}^d(x_i, x_j) = f_d(q_{ij}) \quad (6)$$

In the cost function, φ is composed of a summation at a high level; $\alpha_b, \alpha_d > 0$ are corresponding to the step length of the edges E_b, E_d respectively; each term in the summation is a function of one of the two following similarity measures between the current measurements $\beta(x), d(x)$ and the desired ones β_g, d_g as follow:

$$c_{ij}(x_i, x_j) = \cos(\angle(\beta_{g,ij}, \beta_{ij})) = \beta_{ij} \beta_{g,ij}^T \quad (7)$$

$$q_{ij}(x_i, x_j) = \beta_{g,ij}^T (x_j - x_i - (x_{g,j} - x_{g,i})) = d_i j c_i j - d_{g,ij} \quad (8)$$

Eq. (7) is the cosine of the angle between the measured and desired bearing, so $c_{ij} = 1$ when the bearing coincide; and Eq. (8) quantified the discrepancy between the measured and desired relative position of the agents projected on the line given by $\beta_{g,ij}$, so $q_{ij} = 0$ when bearing and distance coincide, see Figure () for an illustration. In this paper, we use $f_b(c) = 1 - c$ and $f_d(q) = \frac{1}{2} q^2$.

The gradient and control law The gradient of each term in Eq. (9) and Eq. (10) can be computed as:

$$g_{ij}^{b} = -f_b(c_{ij})\beta_{ij} - f_b^{'}(c_{ij})(I_n - \beta_{ij}\beta_{ij}^{T})\beta_{g,ij} \quad (9)$$

$$g_{ij}^{d} = -f_d^{'}(d_{ij}c_{ij} - d_{g,ij})\beta_{g,ij} \quad (10)$$

$$g_i = \alpha_b \sum_{(i,j)\in E_b} g_{ij}^{b}(x_i, x_j) + \alpha_d \sum_{(i,j)\in E_d} g_{ij}^{d}(x_i, x_j) \quad (11)$$

recall we have $f_b(c) = 1 - c$ and $f_d(q) = \frac{1}{2}q^2$, so the control law of Baxter is given by:

$$g^{b} = \beta_g - \beta \quad (12)$$

$$g^{d} = d_g\beta_g - d(\beta^{T}\beta_g)\beta_g \quad (13)$$

$$g_i = \alpha_b g^{b} + \alpha_d g^{d} \quad (14)$$

$$u = -grad\varphi(x) = -g_i \quad (15)$$

Collision avoidance We assume every obstacle is a sphere in this paper, without losing generality, we assume the i_{th} sphere's center located at $c_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T$, and its radius is as long as r_i. In Eq. (15), we generate the control law $u = -g_i$, assume the end of baxter arm is at $s = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}^T$, so the distance from the center of the i_{th} sphere is given by:

$$d_i = \frac{\vec{sc}_i \times u}{||\vec{sc}_i||} \quad (16)$$

If $d_i < r_i$, the Baxter arms will collide with the i_{th} obstacle. We identify the set of such obstacles as $O = \{1, \ldots, M\}$, to avoid collision, we modify u as follow:

$$u^{'} = u + \begin{bmatrix} 0 & 0 & \max_{i\in O}(r_i + z_i) - u[3] \end{bmatrix}^T \quad (17)$$

in which $u[3]$ is the 3_{th} entry of u, so we simply lift the arm by some distance. Of course, we can take other more complex methods, but considering this method is applied on the Baxter robot, taking into account the processing time, refresh rate, hardware conditions and some other constraints, we decided to adopt the simplest obstacle avoidance method, for we think this is one of the most robust method.

2.3. Automation system

This section presents the whole automation system, which is divided into three parts: initial position, object localization and shifting to object space.

Initial position The initial position is arbitrarily determined by the user and the only constraint is the location of the object we want to follow. The goal of this part is to ensure the object is in the field of the Baxter arm camera.

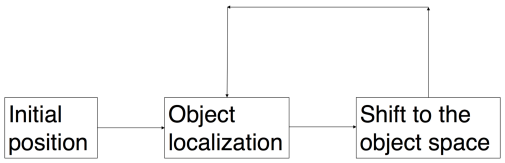

Figure 5. Automation system

Figure 6. General color-based image process

Object localization Using the method described in 2.1, we can locate the object then send its location to the following control model.

Shift to object space Using the method described in 2.2, we can decide the location we want the Baxter moves to.

This automation system is described in Figure 5.

3. Experimental setup

OpenCV OpenCV(Open Source Computer Vision Library) is introduced to use in video image process. General color-based image process is showed in Figure 6.

Generally, color-based detection is sensitive with the surrounding light environment. Through several experiments we did, to get a stable output, a bright and well-lit environment is necessary.

Had appropriate color boundaries is very crucial for getting a right tracking object contour. However, due to the different experimental environment and the particularity of Baxter hand camera , it is hard for us to find a 'constant' color boundary range like pre-experiment using laptop webcam. One feasible method is redefine the range at the start of the experiment. Color boundary used in this experiment is showed in following table.

	$LowerBoundary$	$UpperBoundary$
R	29	84
G	65	175
B	47	255

Baxter hand camera generates relative large noise compared with the laptop webcam, so it is necessary to add filter to have a better output which is sometimes no need for webcam. In general sense of the term "filtering", the image goes through a low pass filter and gets blur to remove the high frequency information which is the detail information of the image. To remove the noise, the value of the filtered image at a given location is a function of the values of the input image in a small neighborhood of the same location, so after filtering, the image resolution will decrease. Generally, gaussian blur is the most popular image blur method. In this case, we would like to keep the edge information when we removed the noise, however, gaussian blur does not preserve edges in the input image. Bilateral filter which is an edge-preserving smoothing filter is be chose. Bilateral filter which first introduced by Tomasi [18] removes fine texture of the image but the overall shading is preserved. Compared with the Gaussian filter, one more (multiplicative) Gaussian filter component which is a function of pixel intensity differences is merged with the original filter.

The build-in function from OpenCV called cv2.bilateralFilter is used in our experiment.

$$cv2.bilateralFilter(src, d, sigmaColor, sigmaSpace)$$

There are four parameters that we can adjust. src: the source image. d: the Diameter of each pixel neighborhood that is used during filtering. sigmaColor: the filter sigma in the color space. sigmaSpace: the filter sigma in the coordinate space. Based on OpenCV docs, it is recommended to use $d = 5$ for real-time applications. For sigma value, 2 sigmas values can be the same. If they are small (less than10), the filter will not have much effect, whereas if they are large (larger then 150), they will have a very strong effect, making the image look cartoonish.

For step five, in Baxter camera environment, contour of the ball cannot be detected fully sometimes mainly because of the color range and this phenomenon will cause the joggle of the tracking target point which is the center of the tennis ball, so we remove the erosion function in order to enlarge the detected pixels to cover the undetected pixels which inside the contour.

After above processes, the performance of the detection is much better than the general one. Figure 7 and Figure 8 show the difference. For each figure, the left one is the HSV image which contains the information (white part) to output the contour of the object and the right one is raw image with marked output center coordinates and radius. Figure 7 shows the output used general image process and Figure 8 shows the output with customized image process based on above analysis. The accuracy of the second one can reach to 0.01 meter after set an appropriate color boundary even in a dark environment.

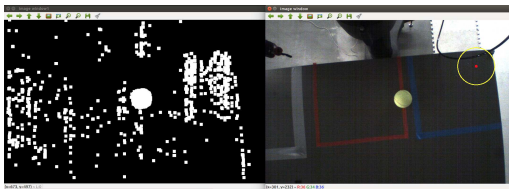

Figure 7. Output of the general process

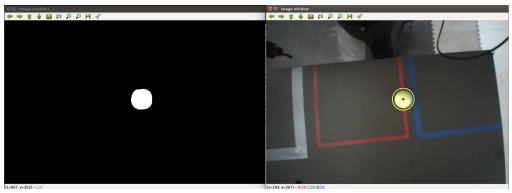

Figure 8. Output of the customized process

ROS Our approach uses ROS(Robot Operating System), which works under Linux and is the default platform for the Baxter robot to work as a middleware for interconnecting all the elements in the proposed system. The communication element in ROS is called "node" which is the ROS term for an executable that is connected to the ROS network. In each node there could have "publishers" and "subscribers" to output and input the information over ROS "topic" element.

In this paper, Baxter default node publishes the raw image of hand camera through "/cameras/left_hand_camera/image" topic. An own created node "trackleft" subscribes the raw image message and use cvbridge which is a ROS library to transform ROS image format to OpenCV format to do above OpenCV image process of each refreshed frame. All these processes work in a callback function of the subscriber. Two publishers which publish the coordinate information of the target points over topic "/Xcoordinate_target" and "/Ycoordinate_target" to the subscribers in the control node to do following movement control action.

4. Results

In this section, we validate our method through simulation on Baxter robot, in this experiment, the object moves on a table, i.e. the object is in 2-D formation, while the Baxter arm is in 3-D formation. The desired formation is set to one where the arm is right on the object, and the distance between them is $0.1m$. The initial positions of both the object and Baxter arm are set randomly. The trajectory of object is shown in red line, while the trajectory of Baxter arm is shown in blue line.

See the result of experiments without obstacle in Figure. 9, and the result of experiments with obstacles in Figure. 10. In each condition, we simulate 3 scenarios. In every

cases, our approach can converge to the desired formation as expected. Finally, when there are obstacles, our approach can make sure Baxter arm can avoid them.

5. Conclusions

In this paper, we proposed an vision servoing approach for 3-D formation control with or without obstacles. This potential function based controller is formed by both the distance measurements and bearing direction measurements, so the main advantage of this controller is more flexible and robust, because we can reduce the dependence on the distance measurements.

However, we also notice that the Baxter arm is not able to converge to the perfect formation perfectly in some cases, besides, the trajectory itself always swings around the object due to the inaccurate object detection. We think the main reason is the constraints derived from Baxter robot, especially the refresh rate. Future work includes more robust object detection as well as experiments in more complicated environment.

References

[1] B. D. O. Anderson, B. Fidan, C. Yu, and D. van der Walle. Uav formation control: Theory and application. *Recent Advance in Learning and Control*, 371:15–34, 2008.

[2] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. *The confluence of vision and control*, 237:66–78, 1998.

[3] F. Chaumette and S. Hutchinson. Visual servo control, part i: Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90, 2006.

[4] P. Corke and S. A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 14(4):507–515, 2001.

[5] A. K. Das, R. Fierro, V. Kumar, J. P. Ostrowski, J. Spletzer, and C. J. Taylor. A vision-based formation control framework. *IEEE Transactions on Robotics and Automation*, 18(5):813–825, 2002.

[6] D. M. Ebert and D. D. Henrich. Safe human-robot-cooperation: Image-based collision detection for industrial robots. *IEEE/RSJ International Conference On Intelligent Robots and Systems*, pages 1826–1831, 2002.

[7] S. A. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, 1996.

[8] Q. Lindsey, D. Mellinger, and V. Kumar. Construction of cubic structures with quadrotor teams. *In Robotics: Science and Systems*, 2011.

[9] E. Malis, F. Chaumette, and S. Boudet. 2.5 d visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250, 1999.

[10] N. Michael, J. Fink, and V. Kumar. Cooperative manipulation and transportation with aerial robots. *Autonomous Robots*, 30(1):73–86, 2011.

[11] N. Moshtagh, N. Michael, A. Jadbabaie, and K. Daniilidis. Visionbased, distributed control laws for motion coordination of nonholonomic robots. *IEEE Transactions on Robotics and Automation*, 25(4):851–860, 2009.

[12] A. Petitti, A. Franchi, D. D. Paola, and A. Rizzo. Decentralized motion control for cooperative manipulation with a team of networked mobile manipulators. *IEEE International Conference Robotics and Automation (ICRA).*, 2016.

[13] R. Robotics. Worked example visual servoing. `http://sdk.rethinkrobotics.com`, 2015.

[14] J. Spletzer, A. K. Das, R. Fierro, C. J. Taylor, V. Kumar, and J. P. Ostrowski. Cooperative localization and control for multi-robot manipulation. *IEEE/RSJ International Conference Intelligent Robots and Systems.*, pages 631–636, 2001.

[15] K. Sreenath and V. Kumar. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots. *Robotics: Science and Systems*, 2013.

[16] K. Stefan and H. Dominik. Fast vision-based minimum distance determination between known and unkown objects. *Intelligent Robots and Systems*, pages 2186–2191, 2007.

[17] M. Stein. Visual servo control of a dual-armed baxter robot. `http://cs.nyu.edu/totok/tpcw.html`, 2014.

[18] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. *Computer Vision, 1998. Sixth International Conference on*, pages 839–846, 1998.

[19] R. Tron, J. Thomas, G. Loianno, K. Daniilidis, and V. Kumar. Bearing-only formation control with auxiliary distance measurements, leaders, and collision avoidance. *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 1806–1813, 2016.

[20] G. A. Vinod, G. Hoang, P. S. Pratama, H. K. Kim, S. B. Kim, and B. H. Jun. Object following control of six-legged robot using kinect camera. *Advances in Computing, Communications and Informatics*, pages 758–764, 2014.

[21] Z. Wang and M. Schwager. Kinematic multi-robot manipulation with no communication using force feedback. *IEEE International Conference Robotics and Automation (ICRA).*, 2016.

[22] W. Xinyu, Y. Chenguang, J. Zhaojie, H. M, and F. Mengyin. Robot manipulator self-identification for surrounding obstacle detection. *Multimedia Tools and Applications*, 76(5):6495–6520, 2017.

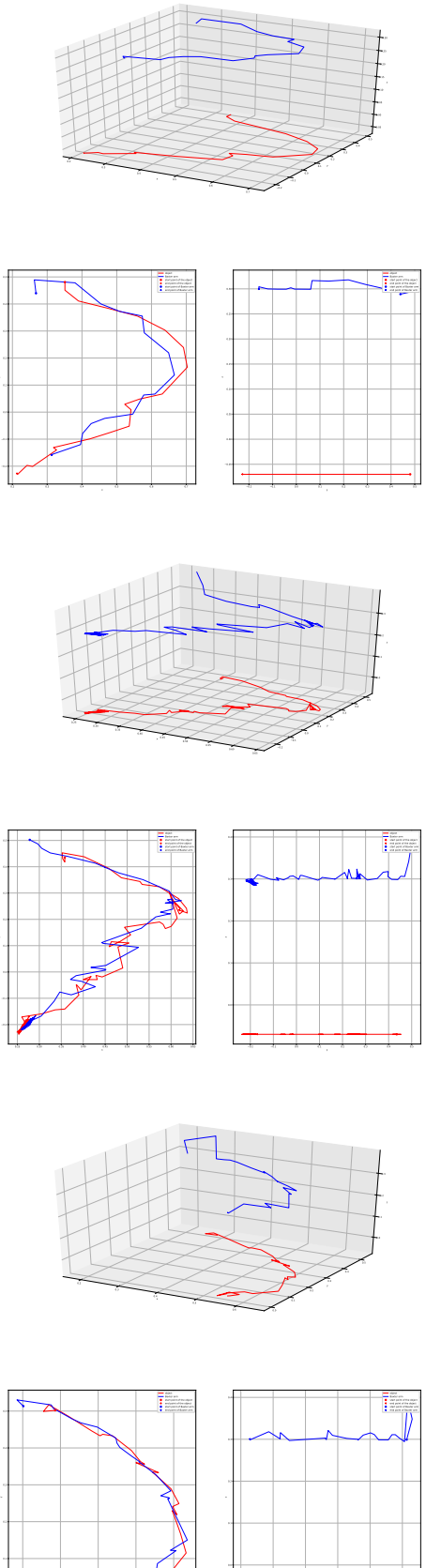

Figure 9. without obstacle

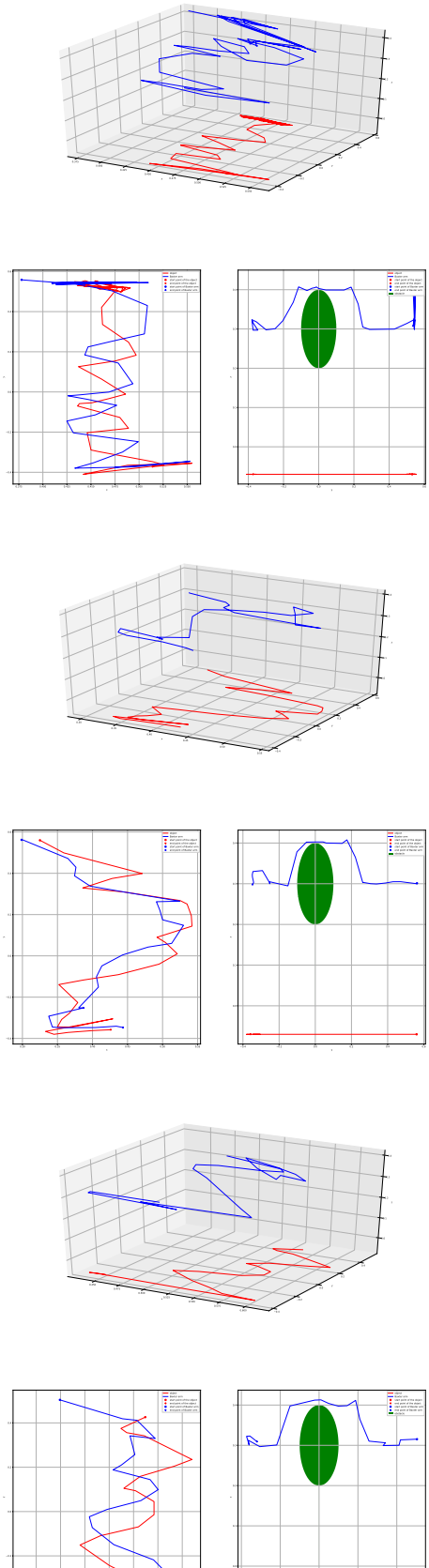

Figure 10. with obstacles