Three Dimensional Tracking of a Submerged Robot

Jeff Dusek

January 17, 2017

1 Introduction

Evaluating the capabilities of an experimental robot design is difficult without a means to objectively evaluate performance. One approach to elucidating the strengths and weaknesses of a design is to analyze how the robot performs in the world using metrics such as speed, endurance, and precision, while simultaneously reviewing the internal states in the control system. By comparing the robots real-world performance with its "beliefs" about the world, iterative improvements can be made in both mechanical and software design.

Motion tracking using single or multiple cameras is a powerful technique for evaluating the performance of a mobile robot. For an underwater robot, visual tracking is complicated by the interface between the air and water mediums, which causes refraction, and subsequent uncertainty as to the position of the object of interest. To accurately track the location of an underwater vehicle in a fluid volume, refraction must be modeled, leading to the use of two camera views to fully rectify the three dimensional robot trajectory.

In the following sections, the general problem of refraction at a flat interface will be introduced and expanded for the case of two perpendicular cameras. The experimental setup used in the underwater robot testing will be shown, along with the techniques for camera calibration in air. Finally, Matlab functions used to optimize setup parameters and solve for the three dimensional coordinates of the robot in world coordinates will be presented and their use explained.

2 Refraction at a Flat Interface

When a ray passes through an interface dividing different mediums, refraction occurs according to Snell's law (1), where n_{medium}, n_{glass}, and n_{air} are the indexes of refraction of the bulk medium (water), the window glass, and air. The angles θ_{medium}, θ_{glass}, and θ_{air} denote the angles of the ray relative to the camera axis in the corresponding media, as shown in Figure 1.

Because $n_{medium} \sim n_{glass}$, the effect on the ray due to the glass is generally omitted, and the problem is considered as a single interface between the air and the bulk medium. In the case of a camera looking into a testing tank, the bulk medium is water with $n_{medium} = n = 4/3$, and $n_{air} = 1$.

$$n_{medium} \sin(\theta_{medium}) = n_{glass} \sin(\theta_{glass}) = n_{air} \sin(\theta_{air}) \tag{1}$$

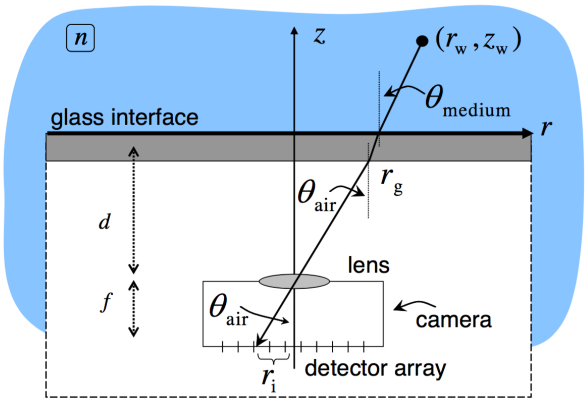


Figure 1: Coordinates and orientation of the camera and interfaces for the case of refraction. Adopted from [2]

2.1 General Case

For the general case of refraction at a flat interface, the $z - axis$ will be considered along the camera axis and assumed to be perpendicular to the interface plane, as shown in Figure 1. Polar coordinates will be used to represent object locations in the image plane and real world respectively as (r_i, ϕ_i) and (r_w, ϕ_w), where r_i is in units of pixels, and r_w is in millimeters. Translation from images coordinates to real world coordinates is also dependent on the focal length of the camera (f), the distance from the entrance pupil to the refraction interface (d), and the distance from the interface to the object of interest (z_w^{obj}).

Taking into account refraction at the air-water interface, the translation from image coordinates (in pixels) to real-world coordinates (in millimeters) is given below (2), as adopted from [2].

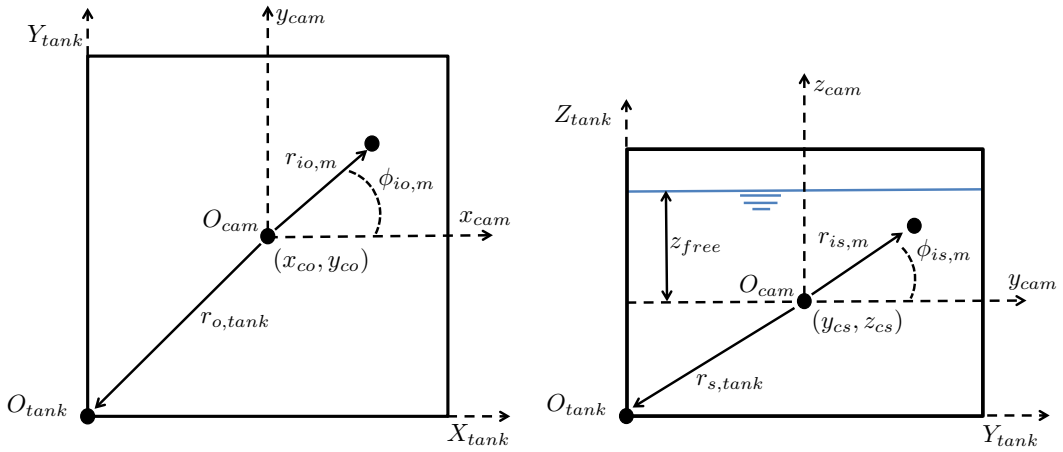
$$r_w = \frac{d}{f} r_i + \frac{z_w^{obj}}{\sqrt{\frac{fn}{r_i}^2 + n^2 - 1}} \tag{2}$$

2.2 Two Camera Views

For the case of tracking a small underwater vehicle in a fluid testing tank, an overhead camera is the primary source of location information. Using a single overhead camera, the distance of the object of interest from the refractive interface (depth) is unknown, making localization of the object in world-coordinates ambiguous. To account for refraction and allow for localization in three dimensions, a second camera is employed to provide a side view of the volume of interest. Both camera views are subject to refraction, and therefore equation (2) must be applied to each view and the equations solved simultaneously.

The coordinate systems and variable names for the two camera views are given in Figure 2. In each case the image coordinate system has it's origin at the image center, which is a property of the camera-lens system and is given as (x_{co}, y_{co}) and (y_{cs}, z_{cs}) for the overhead and side views

respectively. The z-distance of the side camera image center from the free surface, z_{free}, is a property of the experimental setup, and is critical for properly evaluating the depth of the object of interest, and for translating from the camera-frame to the tank-frame.



(a) Imaging coordinate systems from the overhead view.

(b) Imaging coordinate systems from the side view.

Figure 2: Imaging coordinate systems for the overhead and side cameras.

Overhead View The translation from image coordinates to real world coordinates for the overhead camera is given in equations (3) and (6). For the overhead camera view, the distance of the object of interest from the refractive interface ($z_{w,m}^{obj}$) is dependent on the position of the side camera, and the object's location in the side view, as shown in equation (4).

$$r_{wo,m} = \frac{d_o}{f_o} r_{io,m} + \frac{z_{w,m}^{obj}}{\sqrt{\frac{f_o n}{r_{io,m}}^2 + n^2 - 1}} \tag{3}$$

$$z_{w,m}^{obj} = z_{free} - r_{ws,m} \sin(\phi_{s,m}) \tag{4}$$

$$\phi_{is,m} = \phi_{os,m} = \phi_{s,m} \tag{5}$$

$$r_{wo,m} = \frac{d_o}{f_o} r_{io,m} + \frac{z_{free} - r_{ws,m} \sin(\phi_{s,m})}{\sqrt{\frac{f_o n}{r_{io,m}}^2 + n^2 - 1}} \tag{6}$$

Side View For the side view camera, the distance from the refraction interface at the window to the object of interest ($x_{w,m}^{obj}$) is a function of the total tank width (1778 mm), the position of the overhead camera image center, and the position of the object of the interest in the overhead view, as given in equation (8). The translation from image to world coordinates from the side camera is given in equations (7) and (9).

$$r_{ws,m} = \frac{d_s}{f_s} r_{is,m} + \frac{x^{obj}_{w,m}}{\sqrt{\frac{f_s n}{r_{is,m}}^2 + n^2 - 1}} \tag{7}$$

$$x^{obj}_{w,m} = (1778 - r_{wo,tank}\cos(\phi_{wo,tank})) - r_{wo,m}\cos(\phi_{o,m}) \tag{8}$$

$$r_{ws,m} = \frac{d_s}{f_s} r_{is,m} + \frac{(1778 - r_{wo,tank}\cos(\phi_{wo,tank})) - r_{wo,m}\cos(\phi_{o,m})}{\sqrt{\frac{f_s n}{r_{is,m}}^2 + n^2 - 1}} \tag{9}$$

Combined Views To localize the object of interest in three dimensions, equations (6) and (9) for the overhead and side view respectively must be solved simultaneously. In equation (10) the overhead and side view formulations have been combined, and the equation can be solved numerically for $r_{wo,m}$ with the object location in image coordinates ($r_{is,m}$ and $r_{io,m}$) as inputs. With $r_{wo,m}$ known, equation (9) can be used to find $r_{ws,m}$, fully characterizing the object's location in the fluid volume.

$$r_{wo,m} = \frac{d_o}{f_o} r_{io,m} + \frac{z_{free} - \left[\frac{d_s}{f_s} r_{is,m} + \frac{(1778 - r_{wo,tank}\cos(\phi_{wo,tank})) - r_{wo,m}\cos(\phi_{o,m})}{\sqrt{\frac{f_s n}{r_{is,m}}^2 + n^2 - 1}}\right]\sin(\phi_{s,m})}{\sqrt{\frac{f_o n}{r_{io,m}}^2 + n^2 - 1}} \tag{10}$$

Translation to Tank Reference Frame For visualization purposes it is easier to reference the robot position to a reference frame fixed to the tank corner, as shown in Figures 2 and 3. The x and y positions in the tank reference frame are given by equations (11) and (12) respectively. The z-position of robot is chosen to be presented as depth below the free surface in equation (13), as this is a common metric in ocean engineering applications, and allows for direct comparison to depth measured via the onboard pressure sensor.

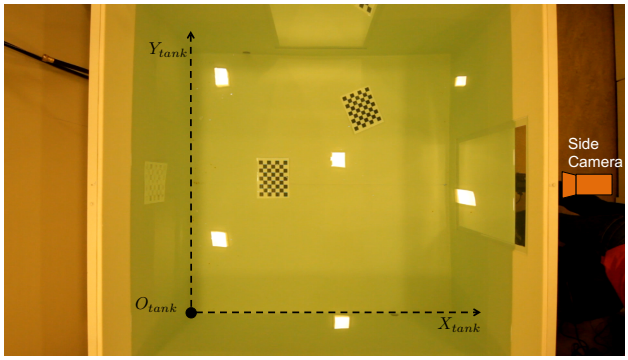


Figure 3: Coordinate system for the tank frame.

$$X_{tank} = r_{wo,m}\cos(\phi_{o,m}) + |r_{o,tank}\cos(\phi_{o,tank})| \tag{11}$$

$$Y_{tank} = r_{wo,m}\sin(\phi_{o,m}) + |r_{o,tank}\sin(\phi_{o,tank})|; \tag{12}$$

$$depth = z_{free} - r_{ws,m}\sin(\phi_{s,m}) \tag{13}$$

3 Camera Setup

To capture the underwater robot from overhead and side views simultaneously, two cameras are positioned outside the water tank. The overhead camera is mounted to an 80/20 frame near ceiling height, as shown shown in Figure 4(a), allowing the entire tank area to be captured in the frame, as seen in Figure 3. The overhead camera is operated remotely from the computer workstation using "Canon's EOS Viewer Utility" software.

The side camera is mounted on a tripod looking through the tank window, as seen in Figure 4(b). Because the width of the window is limited, portions of the tank are occluded from the side camera's view, as illustrated in Figure 5. Because side camera is mounted on a non-permanent tripod, special care must be taken to ensure the camera position and orientation is consistent between experiments. For the tripod used in 4(b), the camera axis is approximately $18\frac{3}{4}$ in or 476.25 mm above the tank bottom.

3.1 Camera Calibration

The overhead and side cameras were each calibrated in air using the Camera Calibration App from the Matlab Image Processing Toolbox [1]. Camera calibration was necessary to find the camera's focal length (f_o and f_s), coordinates of the image center ($x_{co}, y_{co}, y_{cs}, z_{cs}$), and distortion parameters.

The first step in the camera calibration process was to take a series of images using a checkerboard pattern, as seen in Figure 6. When printed on a standard 8.5" x 11" piece of paper, the checkerboard was found to have squares with side length of 28 mm. It is recommended to have a set of approximately 20 images from each camera to ensure accurate calibration.

The Matlab Camera Calibration App provides a convenient GUI for finding the camera parameters, as seen in Figure 7. After loading an image set into the App, corner extraction is carried out automatically, and a calibration can be performed. For the overhead and side cameras, three coefficients were used for the radial distortion, and skew and tangential distortion were omitted, consistent with [2].

The parameters for each camera were exported as *.mat* files, and were used with the Matlab function *undistortImage* to correct camera frames for lens distortion, as seen in Figure 8. For both the overhead and side cameras corrected images were used during the tracking process, although the lens distortion was found to be minimal in both cases.

(a) The overhead camera is a Canon EOS Rebel T5 with a Sigma 8-16 mm Ultra Wide Zoom lens. The camera is mounted above the tank using an 80/20 frame that pivots to allow camera access.

(b) The side camera in a Canon EOS Rebel SL1 with a Sigma 8-16 mm Ultra Wide Zoom lens. The camera is mounted on a tripod and looks through the tank window.

Figure 4: Tracking the underwater robot in three dimensions requires the use of two camera views.

4 Three Dimensional Tracking Code

Several functions have been written in Matlab to extract the position of an underwater vehicle in three-dimensions from two perpendicular camera views as described above. In the following sections, the use of these functions will be outlined. It is assumed that a camera calibration has been performed, and that the parameters have been saved as **cameraParams_overhead** and **cameraParams_side** respectively.

4.1 Convert videos to .avi using FFMPEG

Videos from the overhead and side cameras are saved in .MOV format after acquisition. For use with the tracking code the videos must be converted to .avi using ffmpeg. From the terminal (Mac), navigate to the directory containing the .mov video and use of the following command:

ffmpeg -r 30 -i original.MOV -an -vcodec mjpeg -qscale 2 -r 30 -y output.avi

where *-r* designates the input and output frame rate in frames per second,*-an* removes the audio track, *original.MOV* is the filename of the .MOV file to be converted, and *output.avi* is the filename to be written.

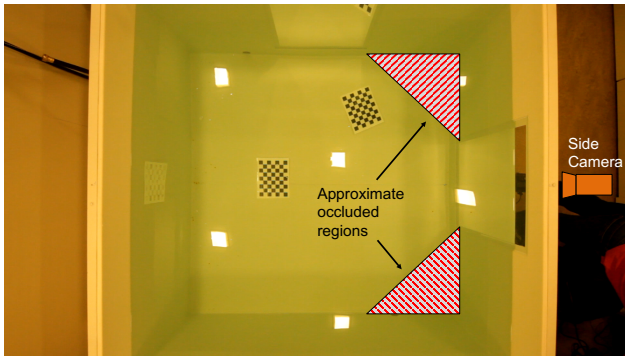


Figure 5: Due to the width of the testing tank windows, potions of the tank are occluded from the side camera view.

(a) Example of images used for calibration of the overhead camera.

(b) Example of images used for calibration of the side camera.

Figure 6: The overhead and side cameras were calibrated by taking a series of images of a checkerboard pattern in air.

4.2 Finding setup parameter d with function *optimizeD*

The distances from the overhead and side cameras to the refraction interfaces, d_o and d_s, are a function of the camera placement and water depth, and should be found for each set of experiments. The function *optimizeD* finds the optimal d values given a single calibration image, such as those shown in Figure 9.

The function call for *optimizeD* is:

[d_opt,error]=OptimizeD(lines)

where the outputs are the optimal value for d and the RMS error in millimeters, and the input *lines* is the number of line segments with known length that will be used to optimize d. When the function is run, the user will be asked which camera is being used, and to select a calibration image.

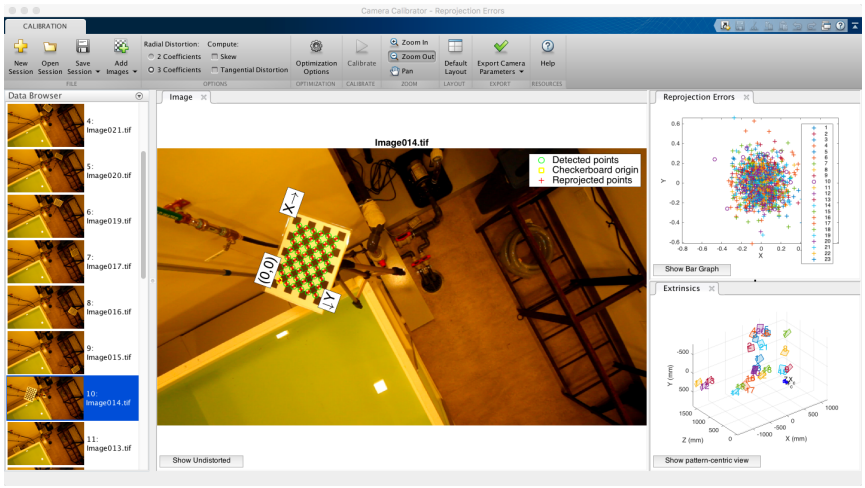


Figure 7: The Image Processing toolbox in Matlab contains a camera calibration app that allows for fast and user-friendly way to find the camera intrinsic parameters.

The calibration image will then be shown as in Figure 9, and the user will manually click on the end points of a line segment of known length. After the endpoints of the line segment are designated, the user must input the true length and depth of the line segment in millimeters in the command window. This process will be repeated for the designated number of line segments. Following the user input of line segments and depths, the optimal value of d will be found using the method outlined in [2].

4.3 Tracking with the function *Videos_to_xyz*

The primary function for tracking the underwater robot is *Videos_to_xyz.m*. Running this function will read in the two camera views, find the image coordinates of the fish centroid, convert to real-world values, and generate a video of the tracking.

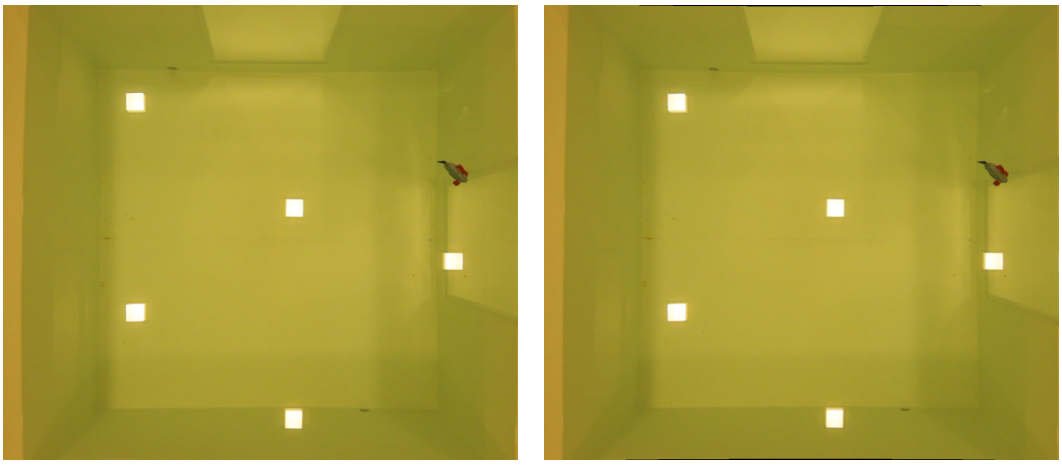
The function call for *Videos_to_xyz.m* is:

[world_outputs]=Videos_to_xyz(start_time_o,end_time_o,start_time_s,end_time_s)

where the output *world_outputs* is a [n x 6] array with the values [x_{cam},y_{cam},z_{cam},X_{tank},Y_{tank},$depth$] as defined in Figure 2. The inputs to the function, *start_time* and *end_time*, are the times to begin and end the tracking in seconds.

After calling the function, the user will be asked to select an overhead video file and to designate the tank region for tracking by selecting four corner points in the image, as shown in Figure 10. After selecting the tank corners in the overhead view, the user will select the side camera video file corresponding to the overhead view.

The result of *Videos_to_xyz* is the array of robot positions in *world_outputs*, a three dimensional plot of the robot trajectory, as shown in Figure 11, and a composite video showing the two camera views with centroid locations and the 3D trajectory. An example frame from the composite video



(a) Raw image frame from the overhead camera before lens distortion has been removed.

(b) Overhead camera image after lens distortion has been removed using the Matlab Image Processing Toolbox. Lens distortion was observed to be minimal for the 8-16mm Ultra Wide zoom lens.

Figure 8: Lens distortion was removed from camera frames in the initial step of the tracking procedure.

is shown in Figure 12.

4.3.1 Setup parameters d_o, d_s, and z_{free}

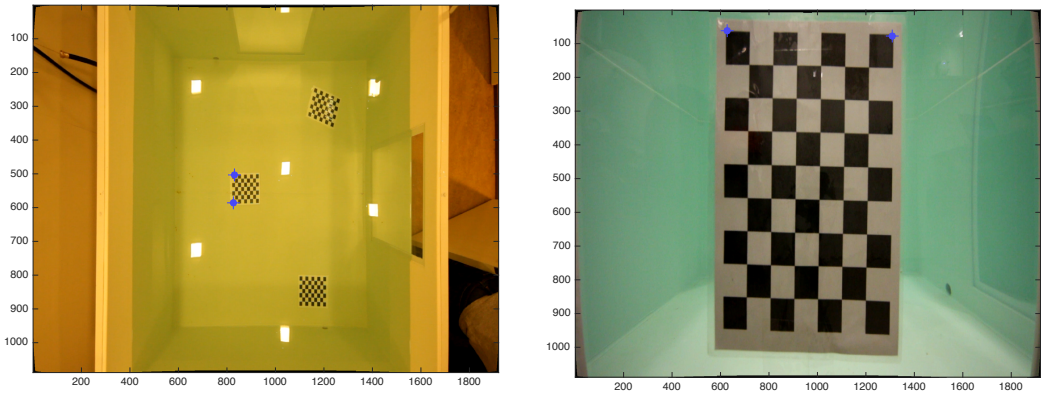
The three setup parameters d_o, d_s, and z_{free} should be updated by the user in *Videos_to_xyz.m* if the experimental setup is changed. The parameters are set in the beginning of the function script, just after the camera parameters are loaded.

4.3.2 Tracking functions used within *Videos_to_xyz*

Two additional functions are used by *Videos_to_xyz*:

Camera_Refraction Takes as inputs the centroid positions from the two camera views in pixels and returns real-world coordinates

Tracking_vids_3D Creates a composite video of the robot tracking with an example frame shown in Figure 12



(a) Overhead calibration image for the setup parameter d_o.

(b) Side calibration image for the setup parameter d_s.

Figure 9: The setup parameter d should be found for each set of experiments to account for variations in camera placement or water depth.

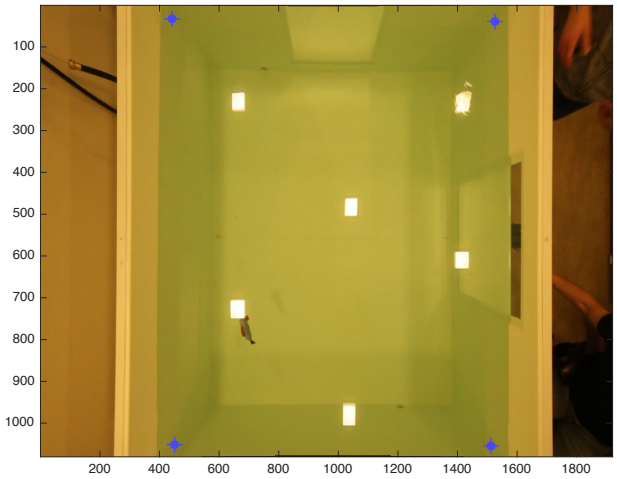


Figure 10: When running *Videos_to_xyz* the user will be asked to designate the tank region in the overhead view to help in the tracking process.

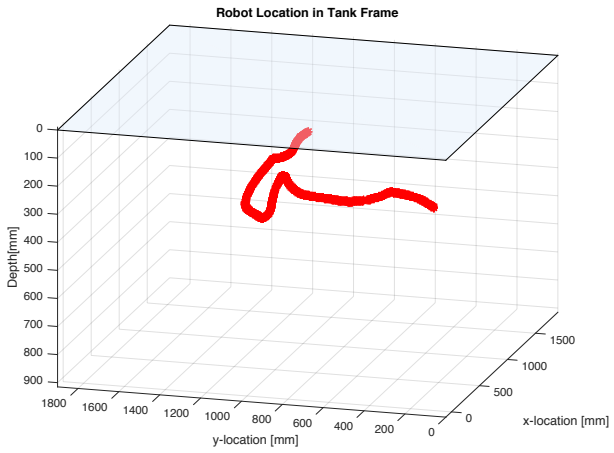


Figure 11: Trajectory of robot in tank frame.

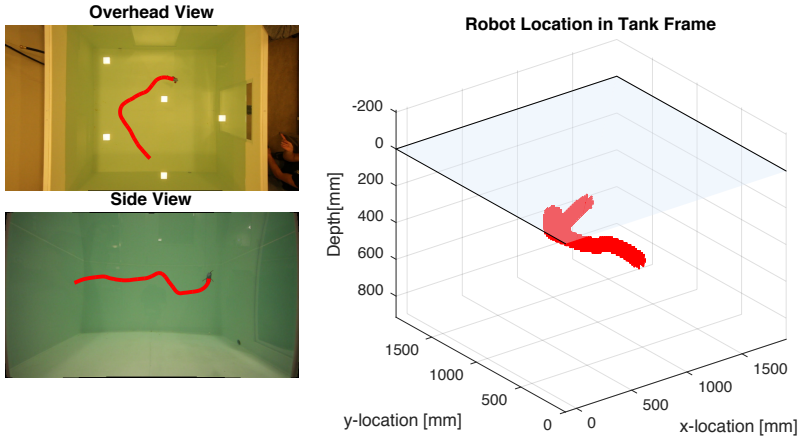


Figure 12: Example frame from the composite video generated by *Tracking_vids_3D* to show the results of the three-dimensional tracking.

Nomenclature

$\phi_{o,m}$ Angular position of object from the camera axis in the overhead view [radians]

$\phi_{s,m}$ Angular position of object from the camera axis in the side view [radians]

$\phi_{wo,tank}$ Angular position of the tank origin from the camera axis in overhead view [radians]

θ_{air} Angle of the ray relative to the z-axis in air [radians]

θ_{glass} Angle of the ray relative to the z-axis in glass [radians]

θ_{medium} Angle of the ray relative to the z-axis in the bulk medium [radians]

d_o The distance from the overhead camera entrance pupil to the interface [millimeters]

d_s The distance from the side camera entrance pupil to the interface [millimeters]

f_o The side camera focal length [pixels]

f_s The overhead camera focal length [pixels]

n_{air} The index of refraction of air

n_{glass} The index of refraction of the tank window glass

n_{medium} The index of refraction of the bulk medium

$r_{io,m}$ The radius from the camera axis to the m_{th} object in the image plane in the overhead view [pixels]

$r_{is,m}$ The radius from the camera axis to the m_{th} object in the image plane in the side view [pixels]

$r_{wo,m}$ The radius from the camera axis to the m_{th} object in the world plane in the overhead view [millimeters]

$r_{wo,tank}$ The radius from the camera axis to the tank origin in world coordinates from the overhead view [millimeters]

$r_{ws,m}$ The radius from the camera axis to the m_{th} object in the world plane in the side view [millimeters]

x_{co} The x-coordinate of the image center from the overhead camera [pixels]

$x_{w,m}^{obj}$ The distance from the interface to the object along the x-axis [millimeters]

y_{co} The y-coordinate of the image center from the overhead camera [pixels]

y_{cs} The y-coordinate of the image center from the side camera [pixels]

z_{cs} The z-coordinate of the image center from the side camera [pixels]

z_{free} The z-distance of the side camera image center from the free surface [millimeters]

$z_{w,m}^{obj}$ The distance from the interface to the object along the z-axis [millimeters]

References

[1] MathWorks. Camera calibrator, December 2016.

[2] Tali Treibitz, Yoav Y Schechner, and Hanumant Singh. Flat refractive geometry. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1–8. IEEE, 2008.