

# Nonlinear Control Framework for Gimbal and Multirotor in Target Tracking

Jae Hun Lee



March 14, 2018

Group objective: To track multiple targets robustly using R-RANSAC.

My objective: To ensure targets are not lost in the camera field of view. Image-based visual servoing (IBVS) provides a good framework.

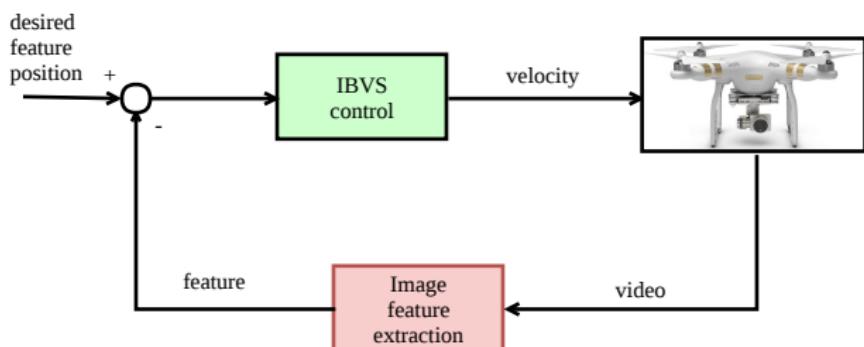


Figure: Image-based visual servoing block diagram

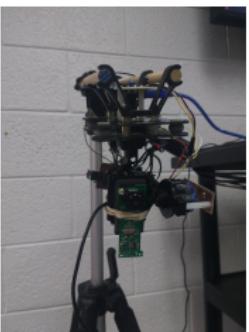
My question: How can I eliminate the needs for the depth term for visual servoing?

## My contributions

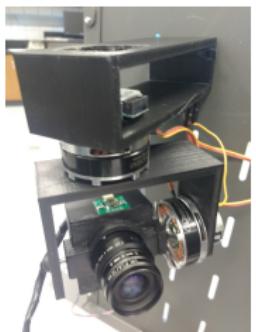
1. presenting three gimbal control algorithms including newly developed adaptive depth gimbal control.
2. integrating the system for multirotor autonomous target following using R-RANSAC tracker and demonstrating the hardware results.
3. developing the unit vector visual servoing framework for UAV control in target tracking.



(a) Ideal



(b) First webcam version

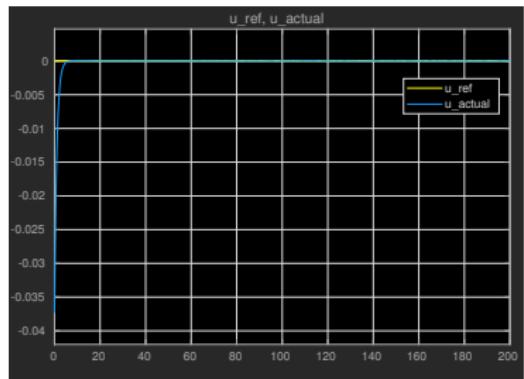


(c) Custom uEYE camera gimbal

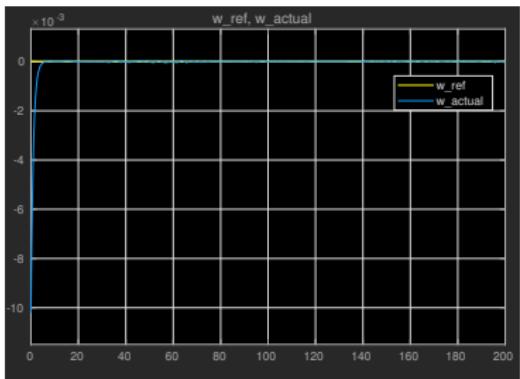
Figure: Gimbal

## Gimbal algorithms

1. Angle commanding gimbal control (UAV book)
2. Angular rate commanding gimbal control (Hurak and Rezac)
  - introduces image jacobian for gimbal control
3. Adaptive depth gimbal control (Me)
  - Model Reference Adaptive Control (MRAC) scheme eliminates the depth term in image jacobian

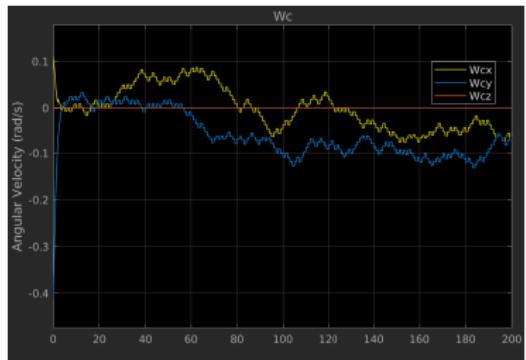


(a) Reference image coordinate  $u_{ref}$  and the system output  $u$

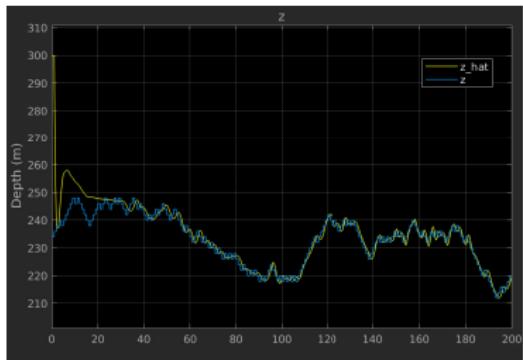


(b) Reference image coordinate  $w_{ref}$  and the system output  $w$

Figure: The simulation result for the adaptive depth gimbal control. The system output  $u$  and  $w$  are converging to the reference model output  $u_{ref}$  and  $w_{ref}$ .

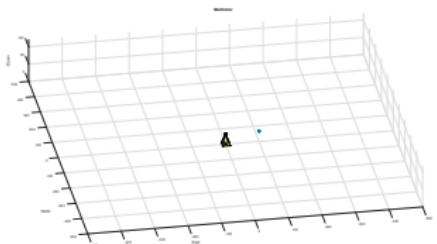


(a) Angular velocity commands

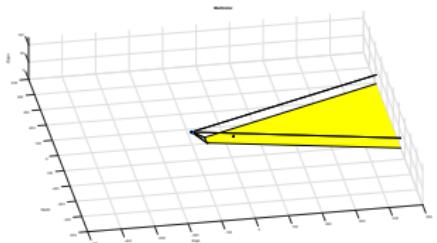


(b) Online depth estimation,  $\hat{z}$

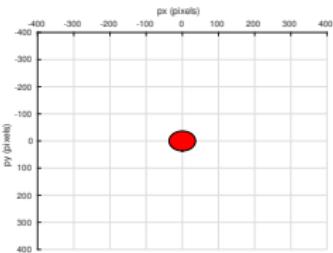
Figure: Angular velocity commands of the adaptive depth gimbal controller. Note that only two commands are used, since it is a pan-tilt gimbal. The depth  $z$  estimate using MRAC.



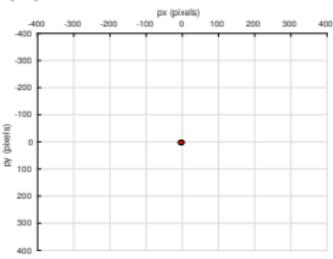
(a) Multirotor at  $t = 0s$ .



(c) Multirotor at  $t = 180s$

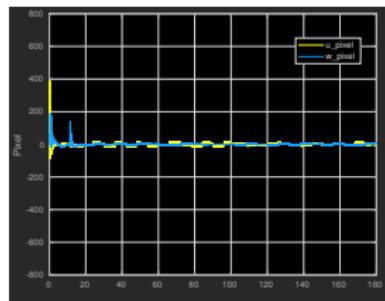
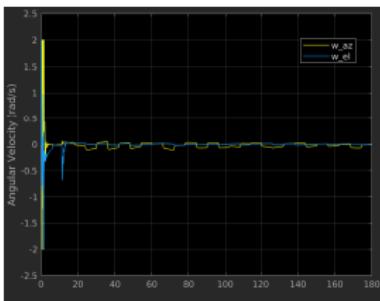
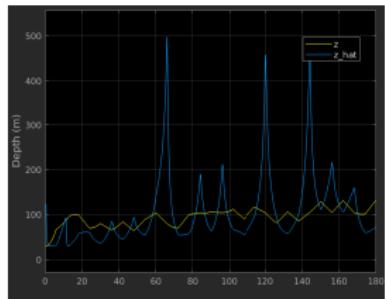


(b) Camera view at  $t = 0s$



(d) Camera view at  $t = 180s$

Figure: Multirotor simulation with camera view. The gimbal pointing objective is well achieved.



(a) Depth  $z$  and its estimate  $\hat{z}$ .  
 Uncertain parameter  
 converging to true value is not  
 guaranteed.

(b) Angular velocity gimbal  
 azimuth and elevation  
 commands

(c) Pixel value  $u$  and  $w$ . They  
 are maintained around the  
 center of the image.

Figure: Uncertain parameter estimation, gimbal angular velocity commands from the controller, and where target lies in the image.

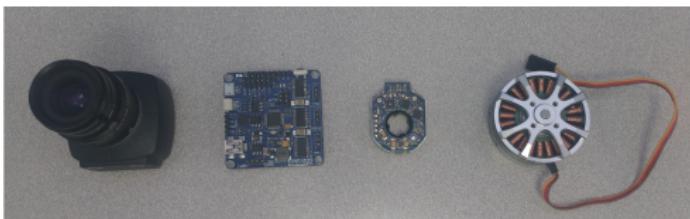


Figure: A custom pan-tilt camera gimbal

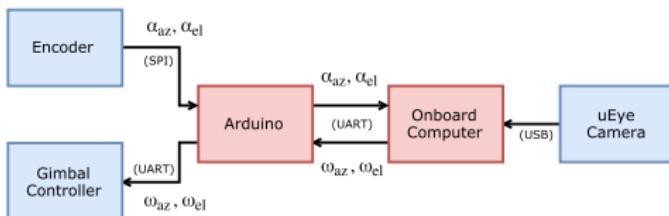


Figure: Custom gimbal block diagram

<https://www.youtube.com/watch?v=VMfvQkhD9-o>

## ACGC

- ▶ Easy to implement
- ▶ Designed for static camera

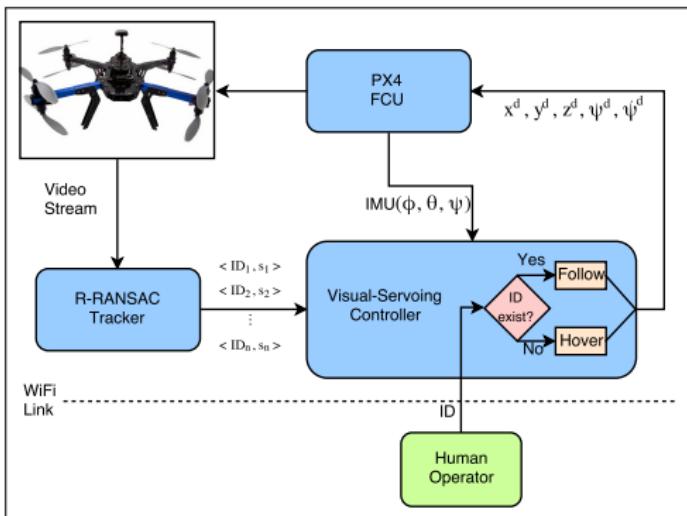
## AVCGC

- ▶ Designed for moving camera
- ▶ Requires the knowledge of the depth  $z$

## ADGC

- ▶ Works without knowing  $z$
- ▶ Suitable for small UAV

## System overview



**Figure: System Architecture.** The R-RANSAC tracker produces a set of target ID numbers and corresponding pixel locations. The visual-servoing controller outputs the desired position, heading, and yaw rate based on the pixel location of the requested target.

Relatively simple controller using camera geometry

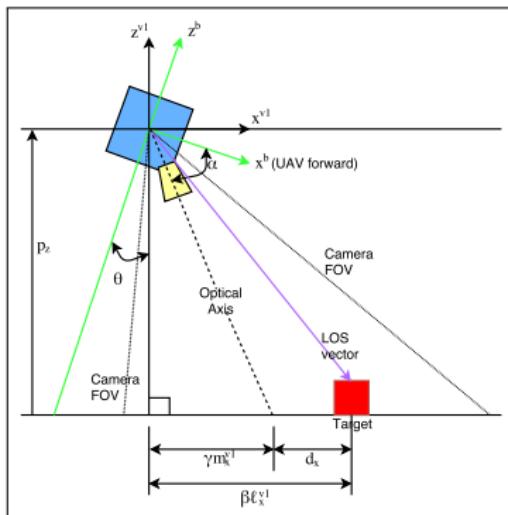
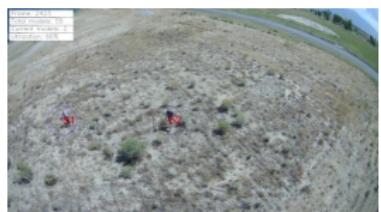
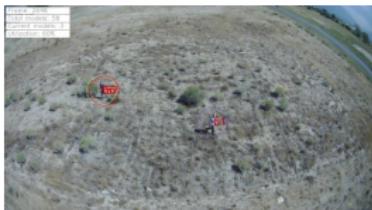


Figure: Side view of the multirotor.

## Hardware results - camera view



(a) Track ID 51 initiated by R-RANSAC tracker ( $t=0s$ )



(b) The human operator has commanded the UAV to follow track ID 51 ( $t=13s$ )



(c) Track ID 65 initiated by R-RANSAC tracker ( $t=27s$ )



(d) The human operator has commanded the UAV to follow track ID 65 ( $t=35s$ )

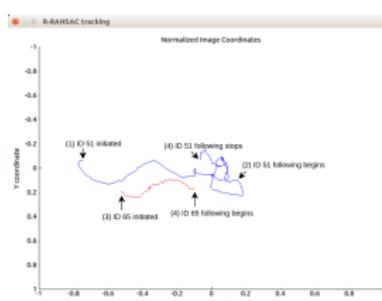


(e) A snapshot of the track ID 65 being followed ( $t=60s$ )

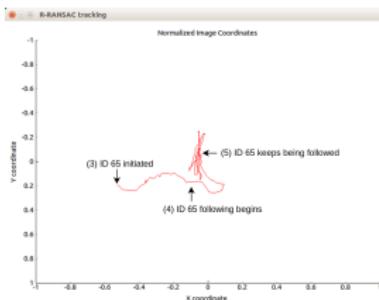
Figure: Camera view at various events

# Autonomous Target Following System

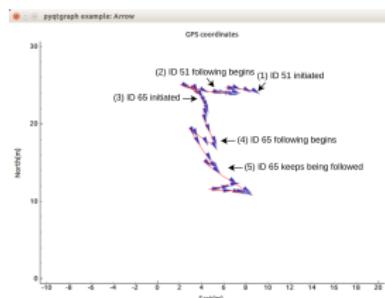
## Hardware results - target footage in image and multirotor GPS footage



(a) Tracks movement in the normalized image plane. Each event (1)-(4) corresponds to camera view in 11a-11d respectively. Until the command to follow ID 65, the multirotor keeps the track ID 51 from leaving the camera view.



(b) The movement of track ID 65 in the normalized image plane. Each event (3)-(5) corresponds to camera view in 11c-11e respectively. The controller keeps the track ID 65 in the camera field of view after receiving the command to do so from the human operator.



(c) Multirotor GPS footage and heading corresponding to camera view in 11a-11e respectively.

## Motivation

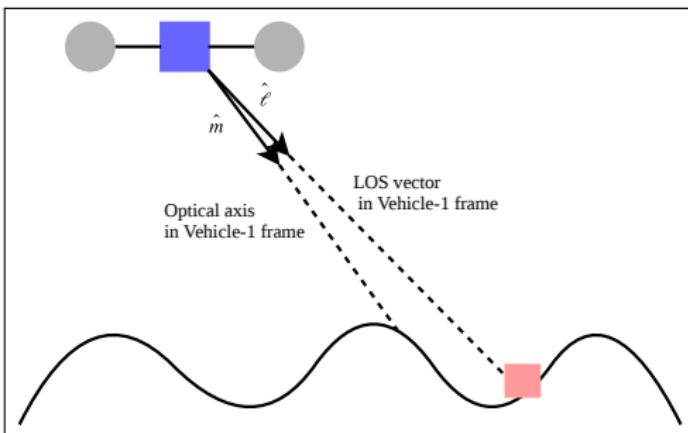


Figure: Non-flat-earth model example. The unit optical axis vector  $\hat{m}$  and the unit line of sight vector  $\hat{\ell}$  are key components of the controller presented in this chapter.

## Control objective

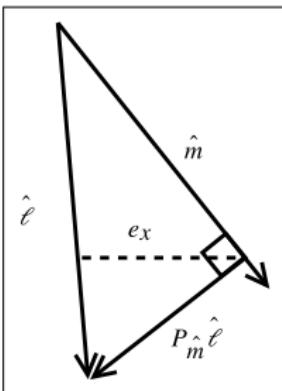


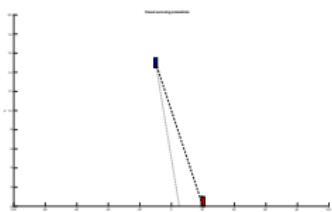
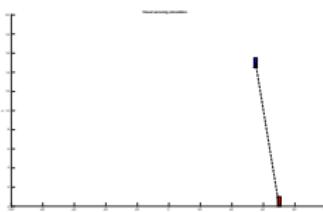
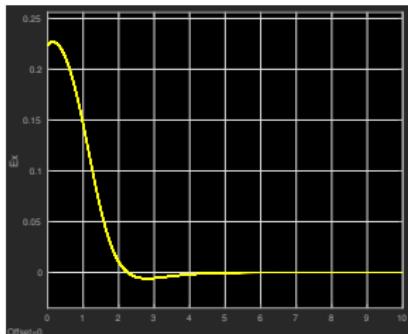
Figure: Projection onto the null space of the optical axis unit vector

$$P_{\hat{m}} = (I - \hat{m}\hat{m}^\top)$$

$$\hat{\mathbf{e}}_1 = [1 \ 0 \ 0]^\top$$

$$\mathbf{e}_x = \hat{\mathbf{e}}_1^\top P_{\hat{m}} \hat{\ell}$$

## Simulation results - simple dynamics

(a) when  $t = 0s$ (b) when  $t = 10s$ 

(c) The horizontal error between the unit LOS vector and the unit optical axis vector converges to zero.

**Figure:** Simple UAV dynamics visual servoing Simulink simulation. The blue square is flying UAV at constant altitude and the red square is a target on the ground moving at  $5m/s$ . The initial UAV and target positions are  $[-10, 15]$  and  $[20, 0]$  respectively. Tuning parameters are set to  $k = 1$ ,  $\Gamma = I_3$  (identity matrix), and  $\alpha = 1000$ .