UAV'S Automatic Landing

2015 June 19 ShanghaiTech University
Robotics Lab Project 2 Presentation

Team member:
Guo Jiangtao
Chen Minhua

Outline

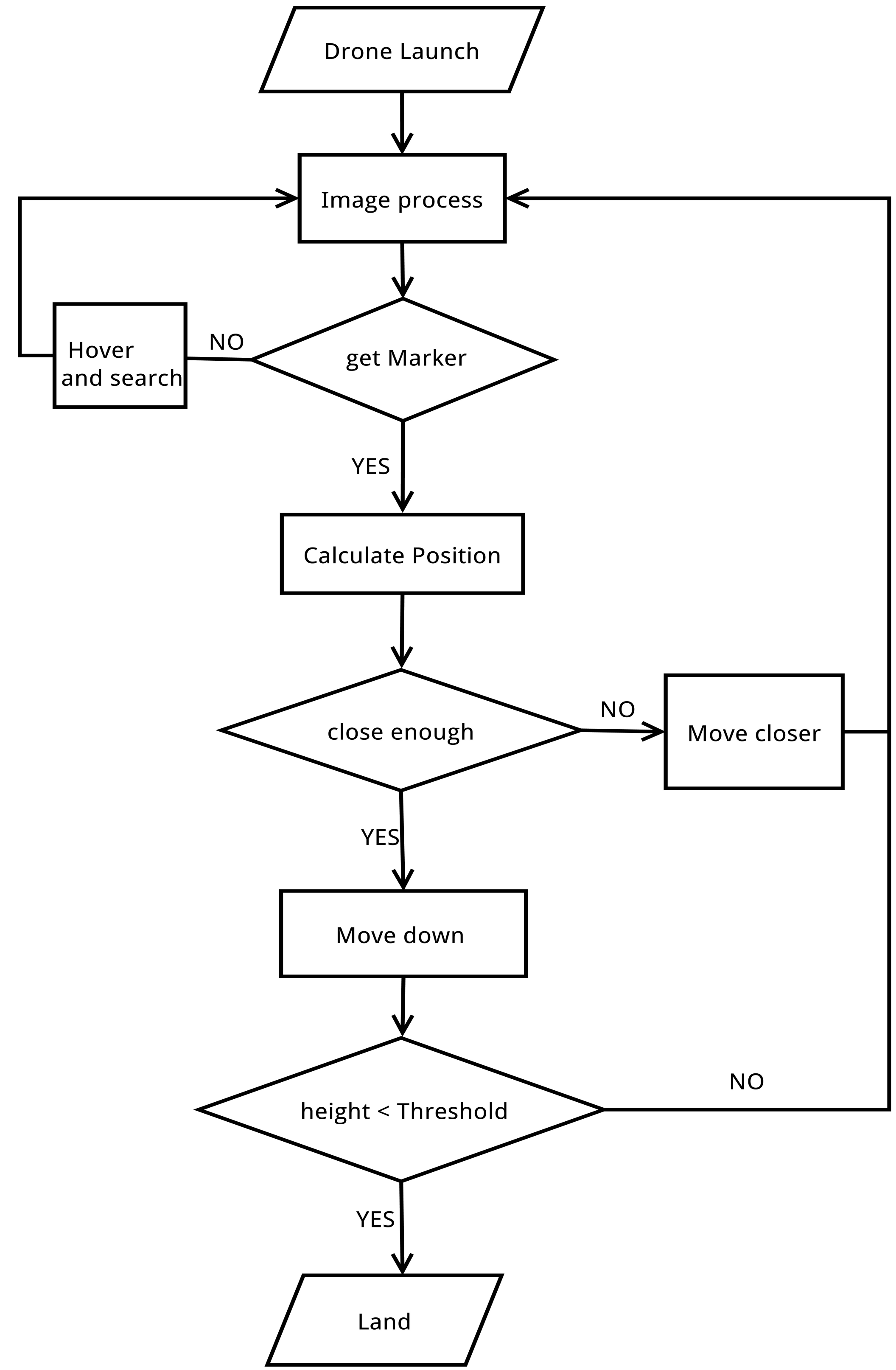
- What we want to do
- Approach
- Result
- Conclusion
- Future Dev

What we want to do

- Land an UAV on a moving target automatically
- Using AR.Drone2.0 as the UAV platform
- Using AR tag marker to mark the target place for UAV to land
- Using PID controller to guarantee stability.
- Using ROS to get the sensor data from UAV and publish command to UAV

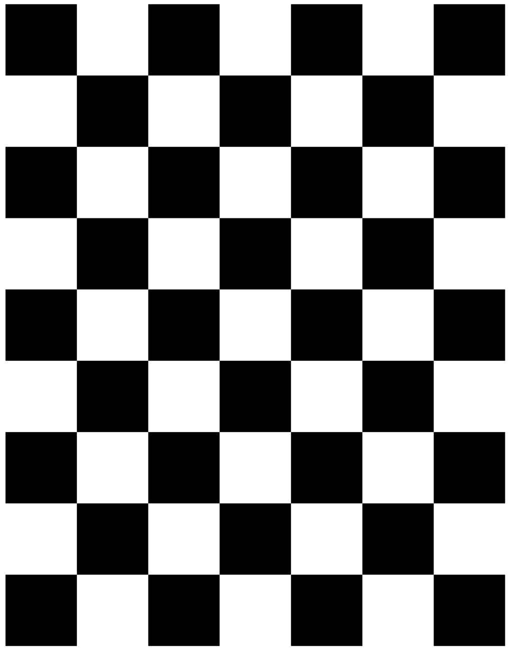
Approach

- camera calibration
- detect the AR tag marker and calculate the position（*http://wiki.ros.org/ar_track_alvar*）
- use pid algorithm to guarantee the stability

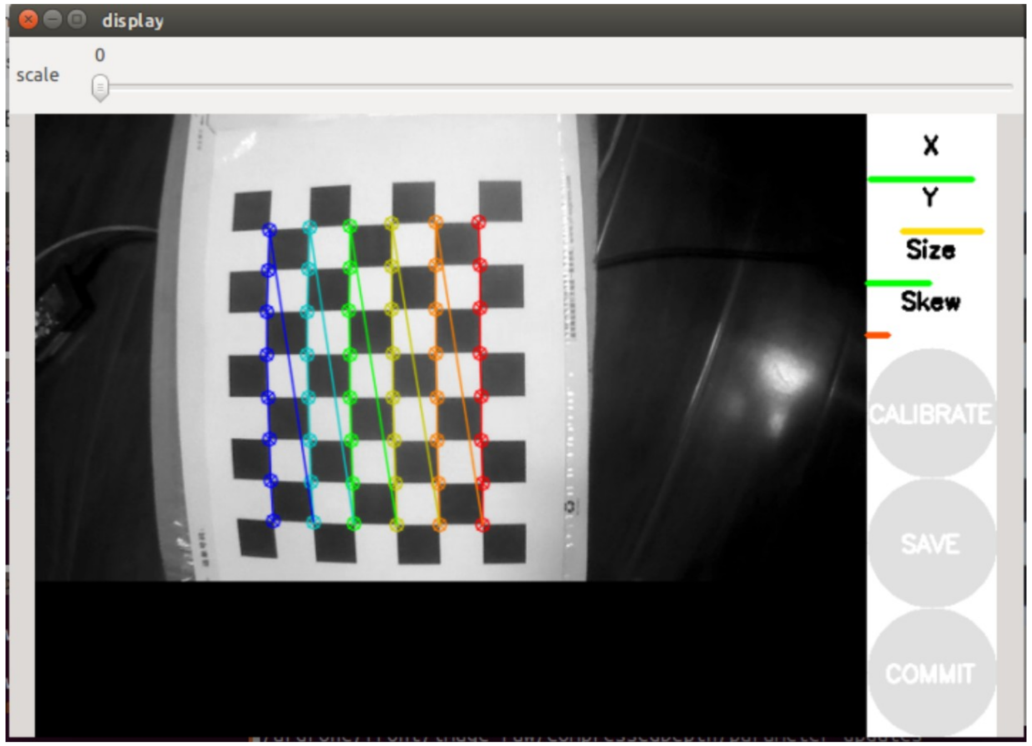


Approach

- **calibration**
- download the checkerboard
- run the camera calibration node($ rosrun camera_calibration cameracalibrator.py --size 8x6 --square 0.0245 image:=/camera/image_raw camera:=/camera)
- move the checkerboard and get 45 samples(different positions)
- get a file with camera's parameter

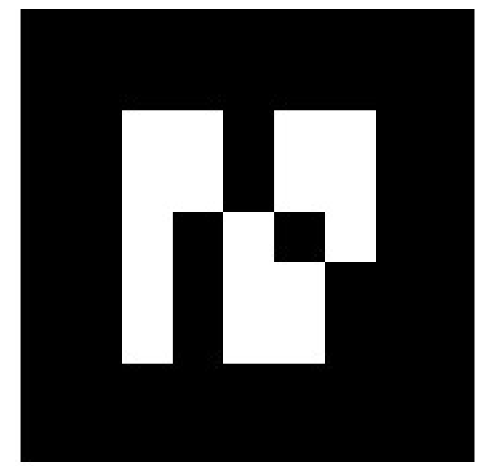



```
image_width: 640
 image_height: 480
 camera_name: camera
 camera_matrix:
   rows: 3
   cols: 3
   data: [640.956, 0, 338.720, 0, 643.880, 227.008, 0, 0, 1]
 distortion_model: plumb_bob
 distortion_coefficients:
   rows: 1
   cols: 5
   data: [-0.134209, 0.195272, 0.000041, -0.00093, 0]
 rectification_matrix:
   rows: 3
   cols: 3
   data: [1, 0, 0, 0, 1, 0, 0, 0, 1]
 projection_matrix:
   rows: 3
   cols: 4
   data: [627.887, 0, 337.724, 0, 0, 632.720, 226.248, 0, 0, 0, 1, 0]
```

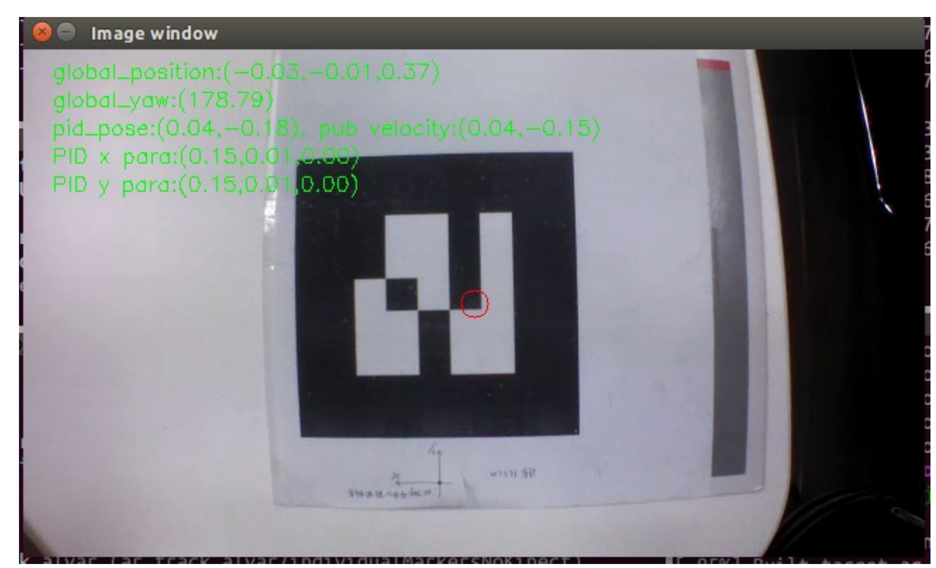



Approach

- **AR tag**



- Augmented Reality markers (13.2cm*13.2cm)
- AR tag Recognition (ar_track_alvar)
- Advantage:
    - fast; easy to get the position (x,y,z wrt camera)
- Drawback
    - contain less information



Approach

- **AR tag**

```
<launch>
    <arg name="marker_size" default="13.2" />
    <arg name="max_new_marker_error" default="0.08" />
    <arg name="max_track_error" default="0.2" />
    <arg name="cam_image_topic" default="/ardrone/front/image_raw" />
    <arg name="cam_info_topic" default="/ardrone/front/camera_info" />
    <arg name="output_frame" default="/ardrone_base_frontcam" />

    <node name="ar_track_alvar" pkg="ar_track_alvar" type="individualMarkersNoKinect" respawn="false" output="screen" args="$(arg marker_size) $(arg max_new_marker_error) $(arg max_track_error) $(arg cam_image_topic) $(arg cam_info_topic) $(arg output_frame)" />
</launch>
```

Approach

- **PID**

- A control loop feedback mechanism (controller) widely used in industrial control systems.

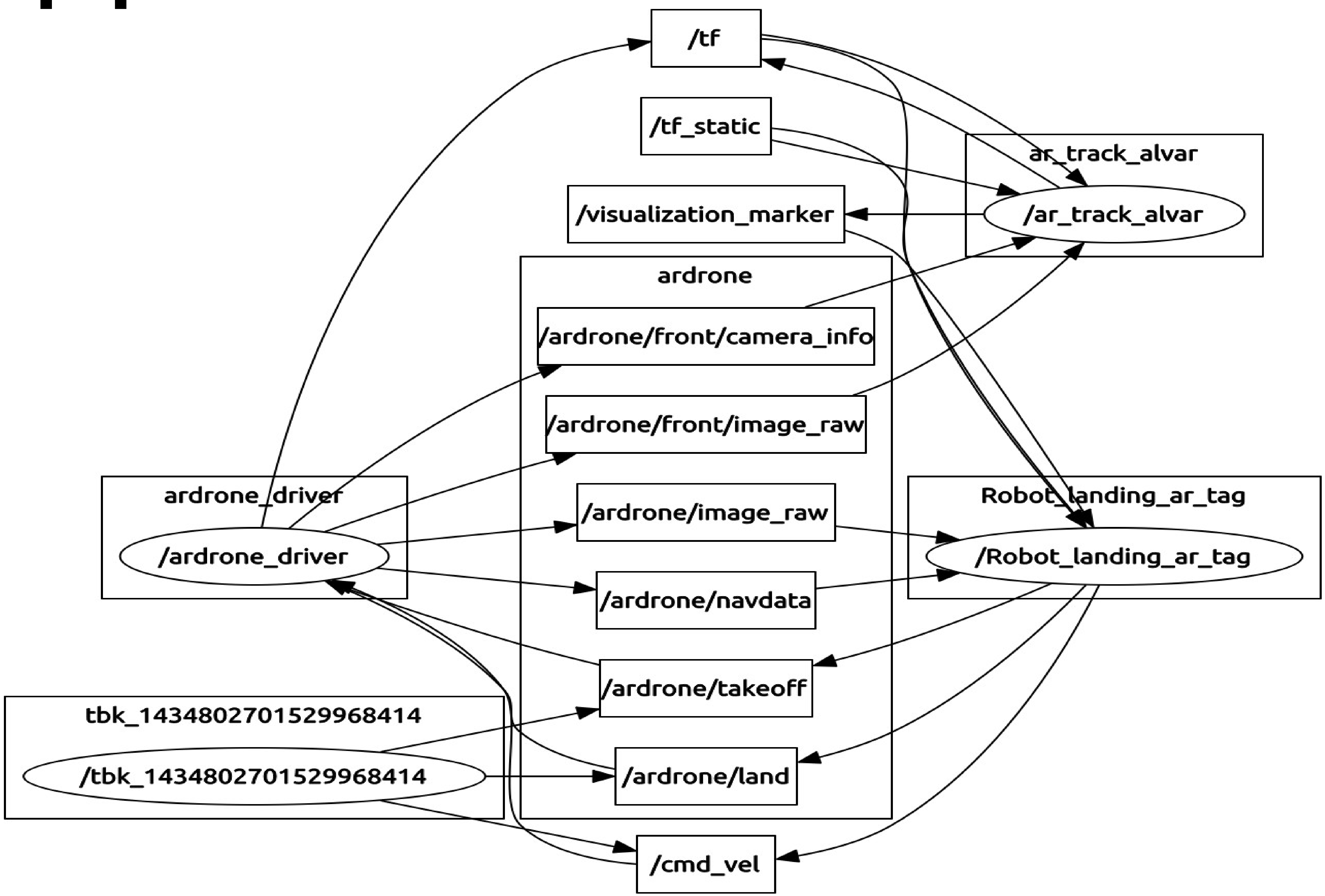
$$U_t = K_P(X_{des} - X_{t-1}) + K_D(\dot{X}_{des} - \dot{X}_{t-1}) + K_I \int_0^t (X_{des} - X_{t'-1})dt'$$

- target position: X_{des} last position: X_{t-1}

- PID parameter: K_P K_D K_I

- speed control command: U_t

Approach


/tf
/tf_static
/visualization_marker
ar_track_alvar
/ar_track_alvar
ardrone
/ardrone/front/camera_info
/ardrone/front/image_raw
/ardrone/image_raw
/ardrone/navdata
/ardrone/takeoff
/ardrone/land
ardrone_driver
/ardrone_driver
Robot_landing_ar_tag
/Robot_landing_ar_tag
tbk_1434802701529968414
/tbk_1434802701529968414
/cmd_vel

Result

- video

Conclusion

- AR tag is faster than QR code, and it’s easier to be detected so that we can get a wider recognisable dynamic range.
- A PID controller calculates an error value as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the error by adjusting the process through use of a manipulated variable. So, PID algorithm can improve the stability.

Future Dev

- We can use a bundle of markers to widen the dynamic range.
- Adjust the PID parameters to improve the stability.
- Improve the accuracy of the position estimation.
- We can use kinect to do the localisation.
- We can use feature matching algorithm to make the UAV move more stable.

Thank you!