PAN-TILT SHOOTER

01- July-2014

PRASHANTH RAMADOSS
VENKATA SAI YESHASVI TIRUPACHURI

Supervisor
GAËTAN GARCIA
Ecole Centrale de Nantes

Objective

- To develop an experimental lab setup for a pan-tilt laser shooter unit to track several light sensitive moving targets
- To implement the setup as an “integration lab” demonstrating several modules like MOCOM, SOFAR, ARPRO and COVIS

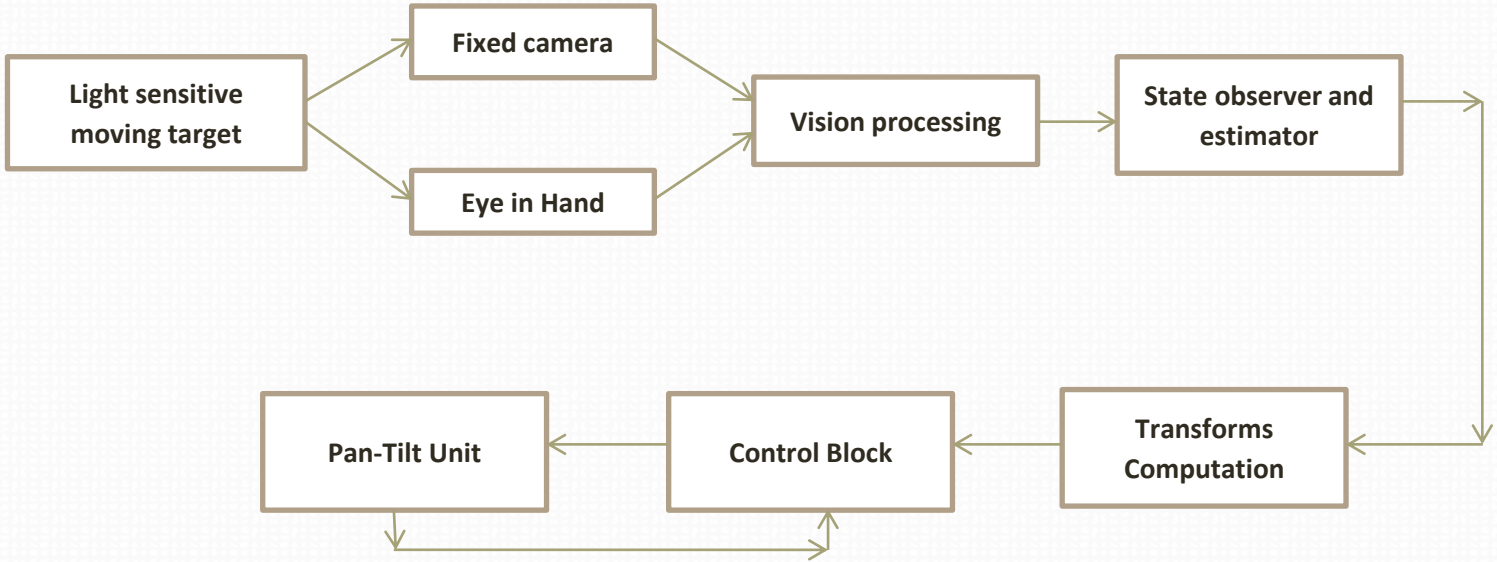
Idea

- To develop modular design of whole application, with fully functional modules
- Integration and Testing of modules done by the students , any module any time
- To provide a fun-filled learning experience

System

- 2-DOF Pan Tilt Manipulator using Dynamixel AX-12A motors
- Laser unit
- Mobile Target
- Light Sensitive circuit counter
- Fixed camera (eye-to-hand config.)
- uEye monochrome camera (eye-in-hand config.)

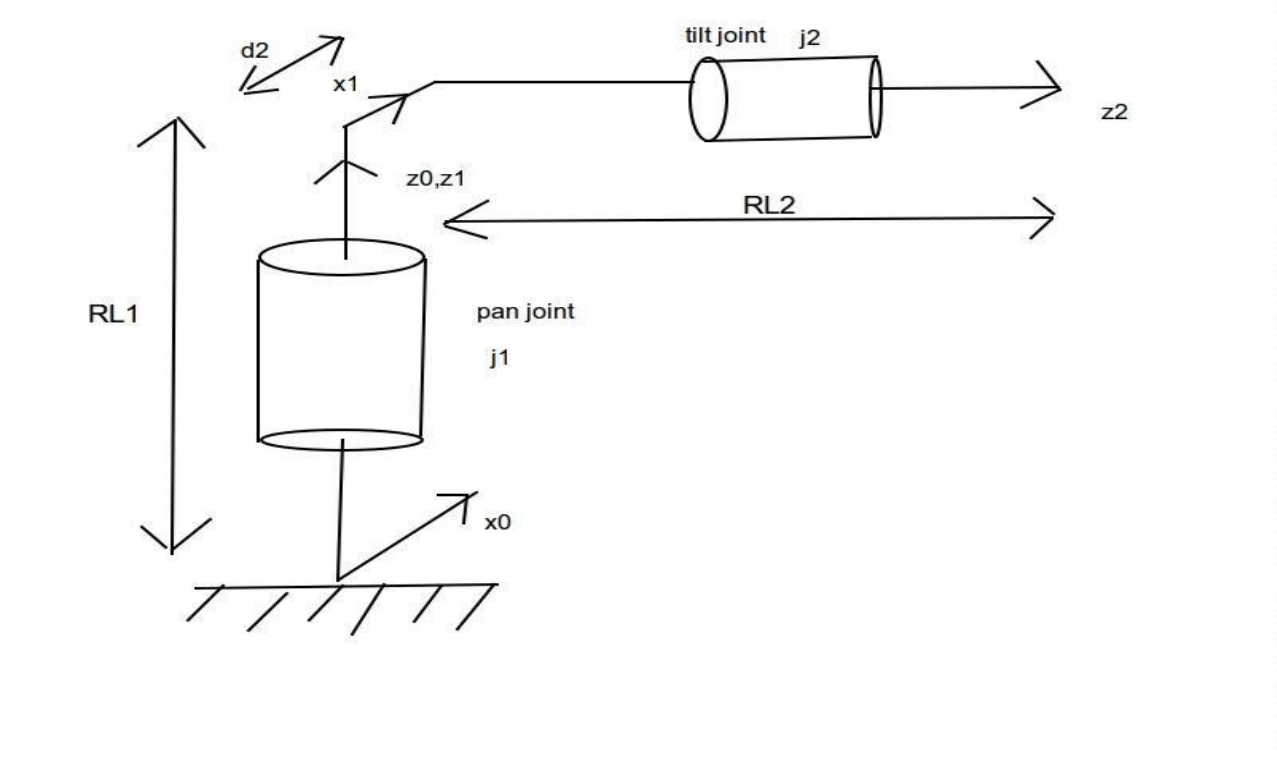
Block Diagram



List of Topics

- Modeling of Robots
- Computer Vision
- C++ Programming
- Software Architecture of Robots
- ROS programming
- Robot Dynamic Identification
- Control (using Computed Torque Control)
- State Estimation

Design and Modeling of Robot



Design and Modeling of Robot

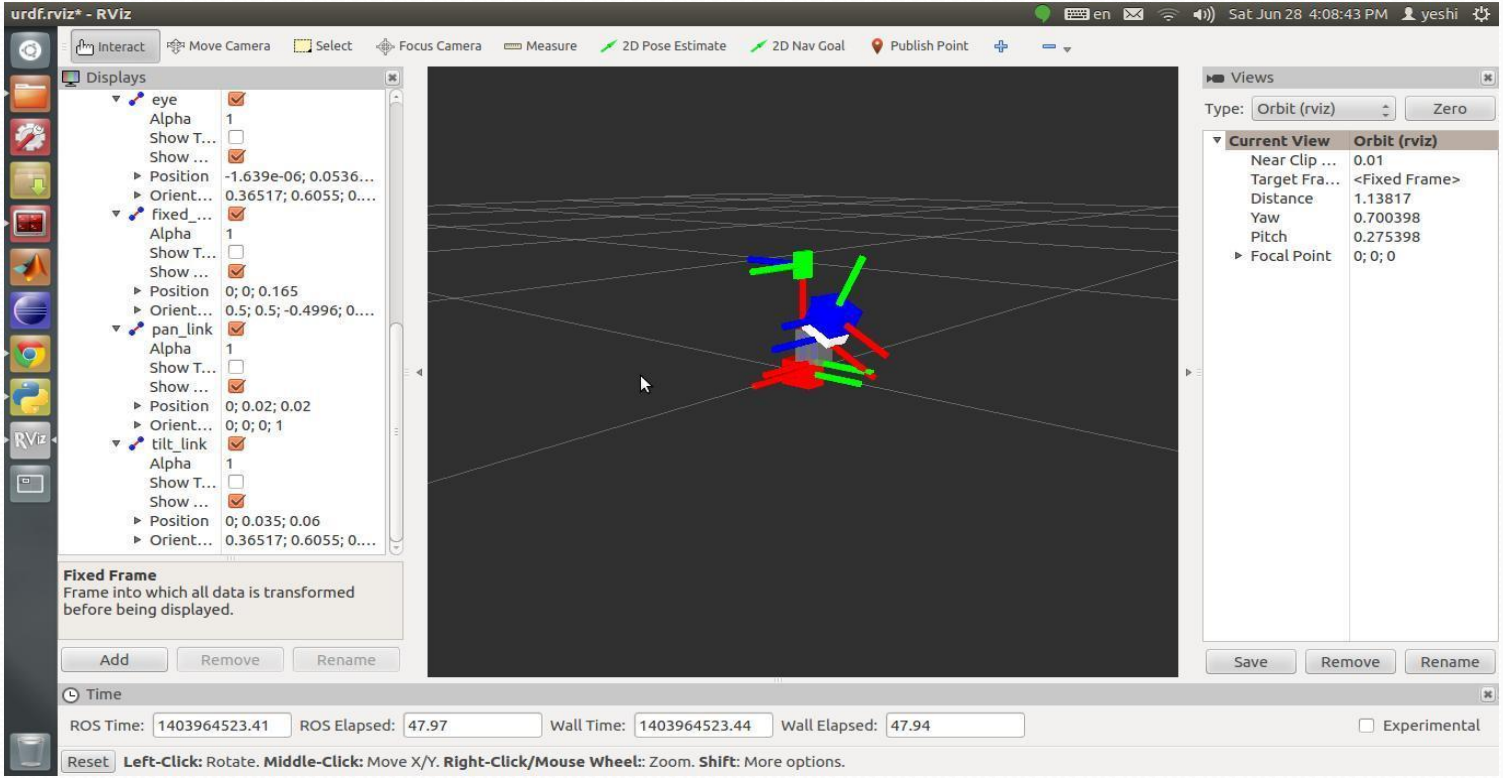
σ_i	j	α_i	$\underset{\sim}{d_j}$	θ_i	$\underset{\sim}{r_j}$	$\underset{\sim}{q_j}$
0	1	0	0	θ_1	RL1	θ_1
0	2	$\pi/2$	d_2	θ_2	RL2	θ_2

Table 1 : Modified DH-parameters

Design and Modeling of Robot

- We developed an URDF (Unified Robot Description Format) describing the links and joints of the robot
- Visualised using RViz Tool

Design and Modeling of Robot



Realisation of Setup

- Actuators used : Dynamixel AX-12A servos

Features of Dynamixel

- Weight: 54.6 g ; Dimensions : 32mm x 50mm x40mm
- Resolution: 0.29 degree;Running Degree: 0 – 300 degree
- Voltage: 9 ~ 12 V
- Command signal: Digital Packet
- Serial Communication (8 bit, 1stop, No Parity)
- TTL interfaces
- ID (0 – 253 IDs)
- Communication speed: 7343 bps ~ 1 Mbps
- Linux SDK: API in C codes

Vision Sensors

- Use of 2 cameras
- Fixed Camera , eye-to-hand configuration, fixed relative to the robot base
- uEye Camera , eye-in-hand configuration, embedded to the end-effector of robot

Features of fixed camera

<<Creative LIVE! CAM Chat HD>>

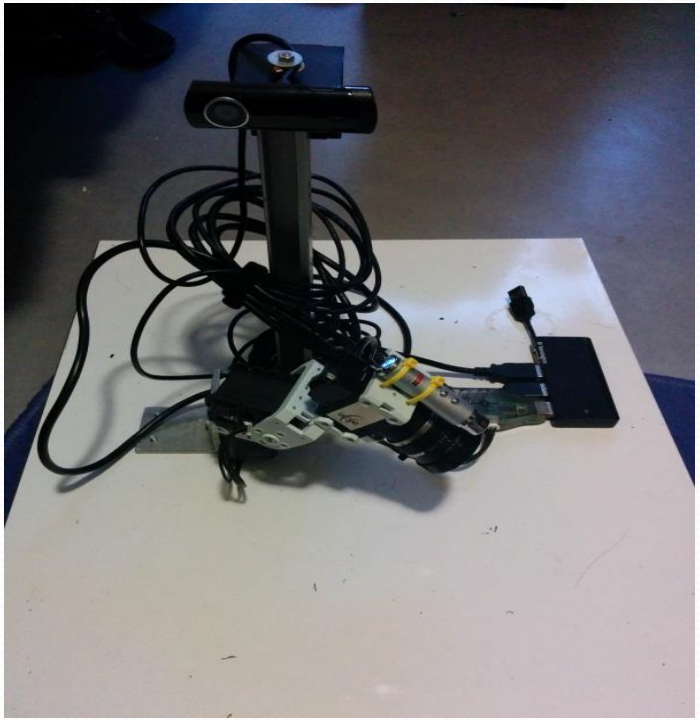
- Resolution : 1280 x 720 (0.9 megapixels)
- High quality CMOS sensor
- Frame rate : up to 30 frames/ s
- Camera control via USB 2.0
- Windows, Linux compatible
- ROS package : uvc_camera node in camera_umd stack

Features of uEye camera

<<IDS uEye 1210-M>>

- Resolution : 640 x 480 pixels (VGA)
- High quality CMOS sensors
- Frame rate : up to 55 frames/ s in full frame mode ; 100 frames/s using Area of Interest
- Small size 34 x 32 x 27.4 (H x W x D), 62 g
- Camera control via USB 2.0
- Linux and Windows compatible
- ROS package : ueye_cam

Implemented Setup



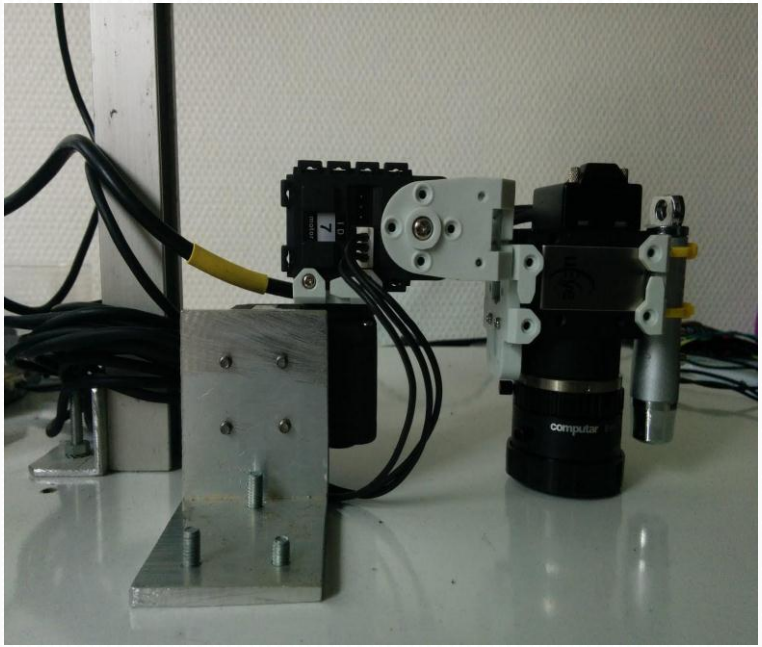
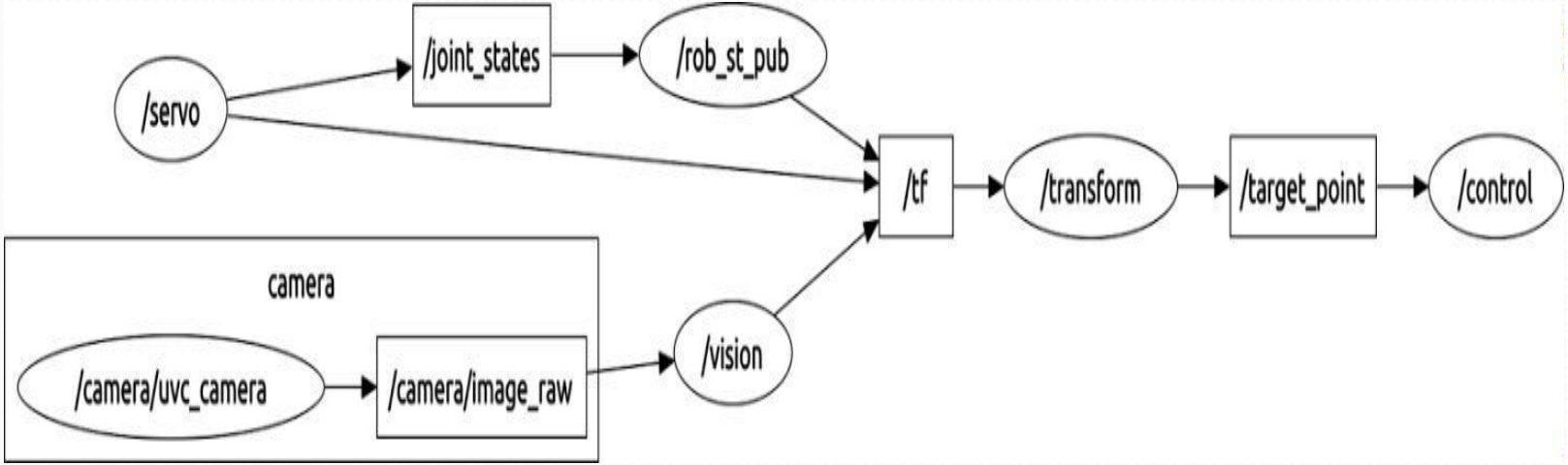


Fig : a) The complete Setup

b) Home position of the Pan-Tilt Unit

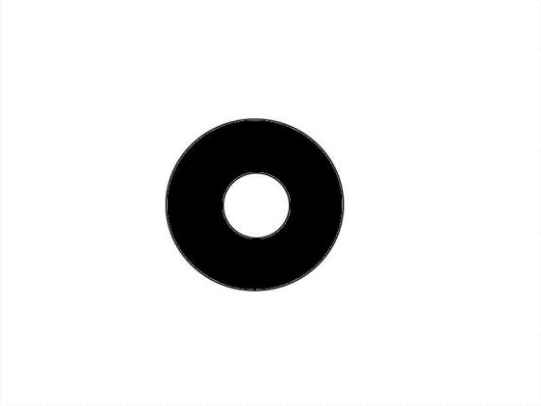
Node Structure



Description of Targets

Target 1

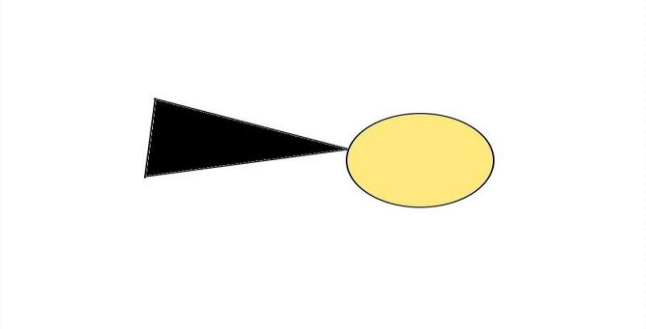
- Binary target
- Required Information : position



Description of Targets

Target 2

- ROI : binary, non-symmetrical
- Required information : Position, orientation (and eccentricity , optional)



Description of Targets

Target 3

- Texture based target , rich of features and interest points

Light Sensitive Circuit on the target

The module consists,

- Laser diode with a defocusing lens, activated by digital signals
- Photodiode whose transient currents change when the light from the laser-diode falls on it
- Counter and Seven-segment LED

Processing the Images

- uEye Camera : 8 bit single channel images, grayscale, passed as CV_8UC1 in OpenCV, sensor encodings MONO8 in ROS
- Fixed Camera : 8 bit three channel images, color images, passed as CV_8UC3 in OpenCV, sensor encodings BGR8 in ROS
- CvBridge to convert ROS image messages to OpenCV images and vice-versa

Techniques used

- Color based Tracking, using Image moments, simple and computationally fast
- Shape based Tracking, using Hough circles and contours
- Feature based Tracking, using Speeded-Up Robust Features

Color Based Tracking

- Easiest way to detect and segment an object
- Simple Thresholding
- Convert the BGR images to either HSV or gray-scale and perform thresholding
- Noisy patches or holes removed by morphological opening and closing

Color Based Tracking

- Position and Orientation using Image moments
- The Moment M_{ij} of a patch in an image is given by

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y)$$

Color Based Tracking

- OpenCV function : Moments moments(thresholded_image)
- Area of the patch : Zeroth order Central moments
- Position of centroid (x,y) : $\{M_{10}/M_{00}, M_{01}/M_{00}\}$

Color Based Tracking

Orientation :

$$\Theta = \frac{1}{2}\arctan\left(\frac{2\mu'_{11}}{\mu'_{20} - \mu'_{02}}\right)$$

Major and minor axes lengths :

$$\lambda_i = \frac{\mu'_{20} + \mu'_{02}}{2} \pm \frac{\sqrt{4\mu'^2_{11} + (\mu'_{20} - \mu'_{02})^2}}{2},$$

using second order central moments

Drawbacks of Color Based Tracking

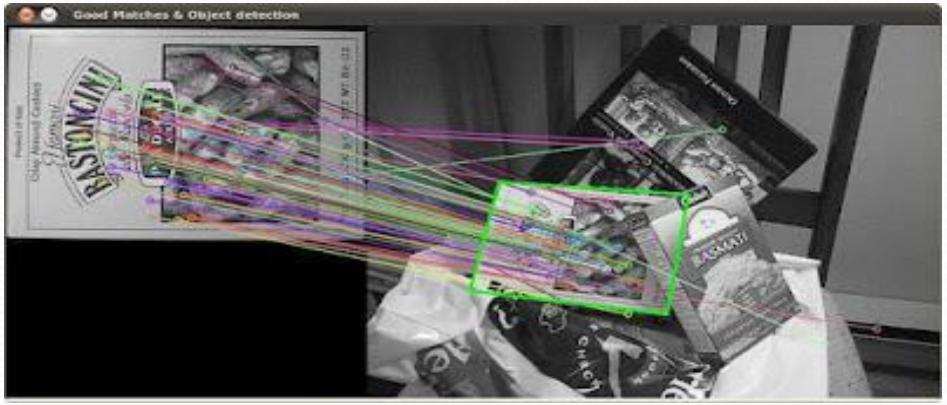
- Improper Lighting Conditions -> Bad tracking
- Existence of more than one object shifts the centroid -> Bad tracking

Shaped Based Tracking : Hough Circles for target 1 or Contour based tracking for complex targets

Texture Based Tracking

- Speeded-Up Robust Features
- Scale Invariance and Rotational Invariance
- 64 dimensional description vector
- Outline : Compare and Match features of Query images with Training images
- 3 stages : Salient Point Detection , Description and Matching

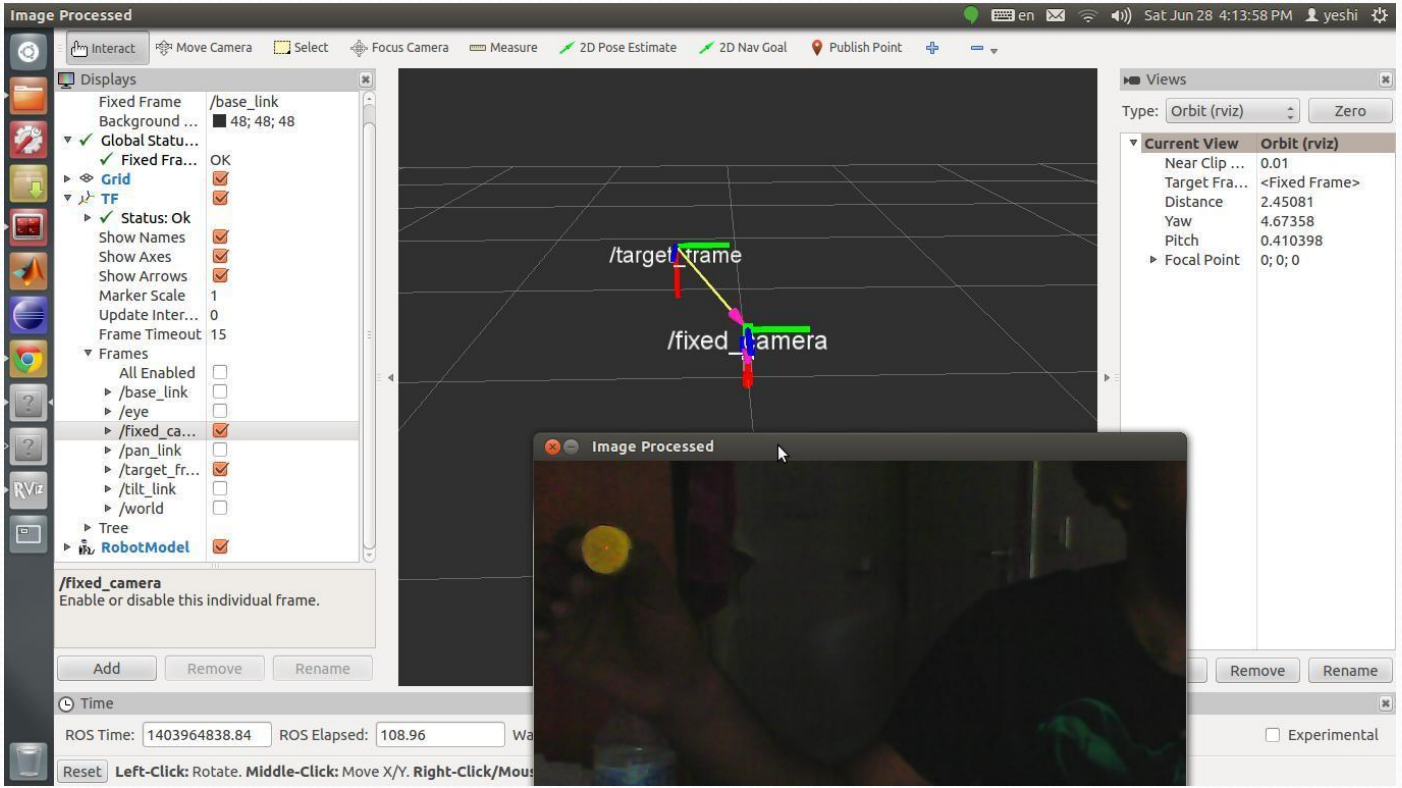
Texture Based Tracking



Drawbacks :

- It is not real time efficient. Use of estimators ??
- Object detection reduces with distance
- If the frame-rate is not properly set, then it may result in false recognitions

Visualisation of Tracking



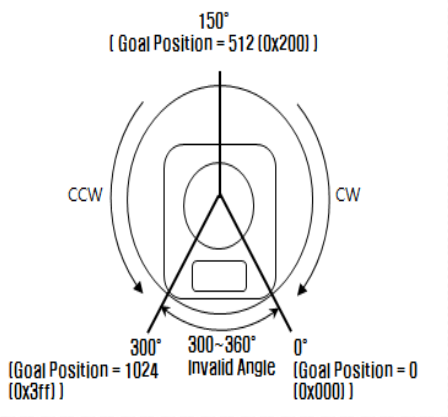
Calibrations and Unit Conversions

- Dynamixel Calibration
- Fixed Camera Calibration

Dynamixel Calibration

- Angle ranges -150 degree to 150 degrees over 1024 steps
- Angle ranges in radians -2.61 to 2.61
- For example consider a servo position of 400

Joint Angle = [(400/512)*2.61]-2.61
= 2.04 - 2.61
= -0.57



Fixed Camera Calibration

- 640 X 480 Resolution
- Fixed camera frame is centered at the image frame
- For rviz visualization multiplied with a factor of 0.001
- So the effective width range is -0.32 to 0.32 and effective height range is -0.24 to 0.24

Obtaining Joint Values

- For pan joint the range in radians is
  = 24.8 (half of 49.6)/150 * 2.61
  = 0.43

- Let's say x – coordinate = 0.1
- Radians corresponding to this position = (0.1/0.32) * 0.43
  = 0.134

- Pan joint servo position = (0.134/2.61)*512
  = 26

- But the true value of the joint is obtained by adding 512 with the above value

Live Demonstration

Results

- Good setup to explain techniques of modeling and vision using C++ and ROS concepts
- Testing robustness of Vision algorithms with simple tracking control

Further Extensions

- Extended Kalman Filtering for Position Estimation (Image co-ordinates)
- Dynamic Modeling
- Computed Torque Control

Thank you