IVR Assignment

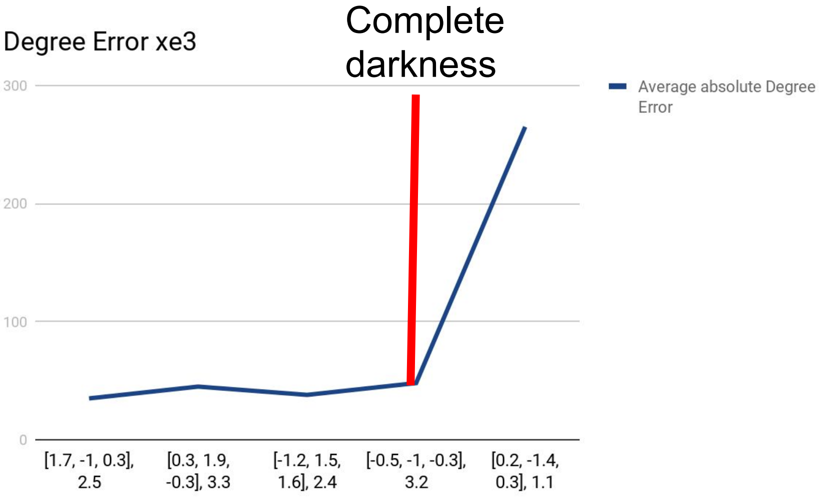
Kieran Litschel, Frederik Kelbel

A - Robot Vision - Joint State Estimation

Method:

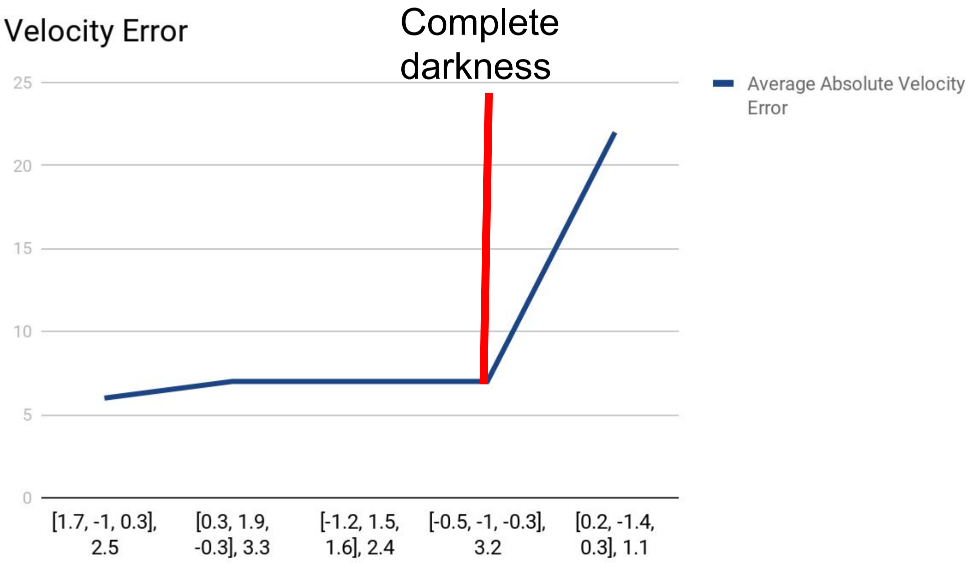
- Find threshold image for each joint using inRange to select range of values each joint could have
  - For third and fourth joint use luminosity to vary the range
- Find centre of joint in threshold image using templates for each joint to find most likely position of centre
  - Necessary to handle overlapping joints
- Calculate joint angles using rotation matrices

Average Error over 5 target hits



Degree Error xe3

Complete darkness

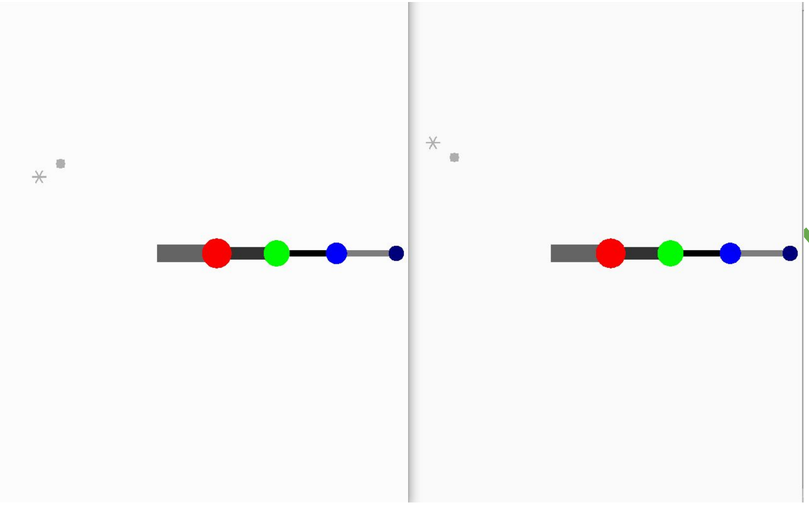


Velocity Error

Complete darkness

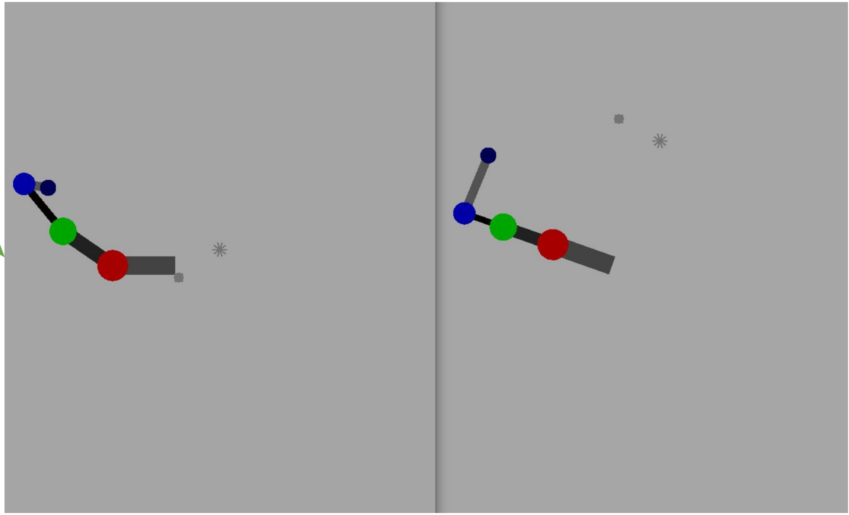
Example Run (TORQUE)

xy-plane xz-plane



Ee-distance to target: 6.379291843349659

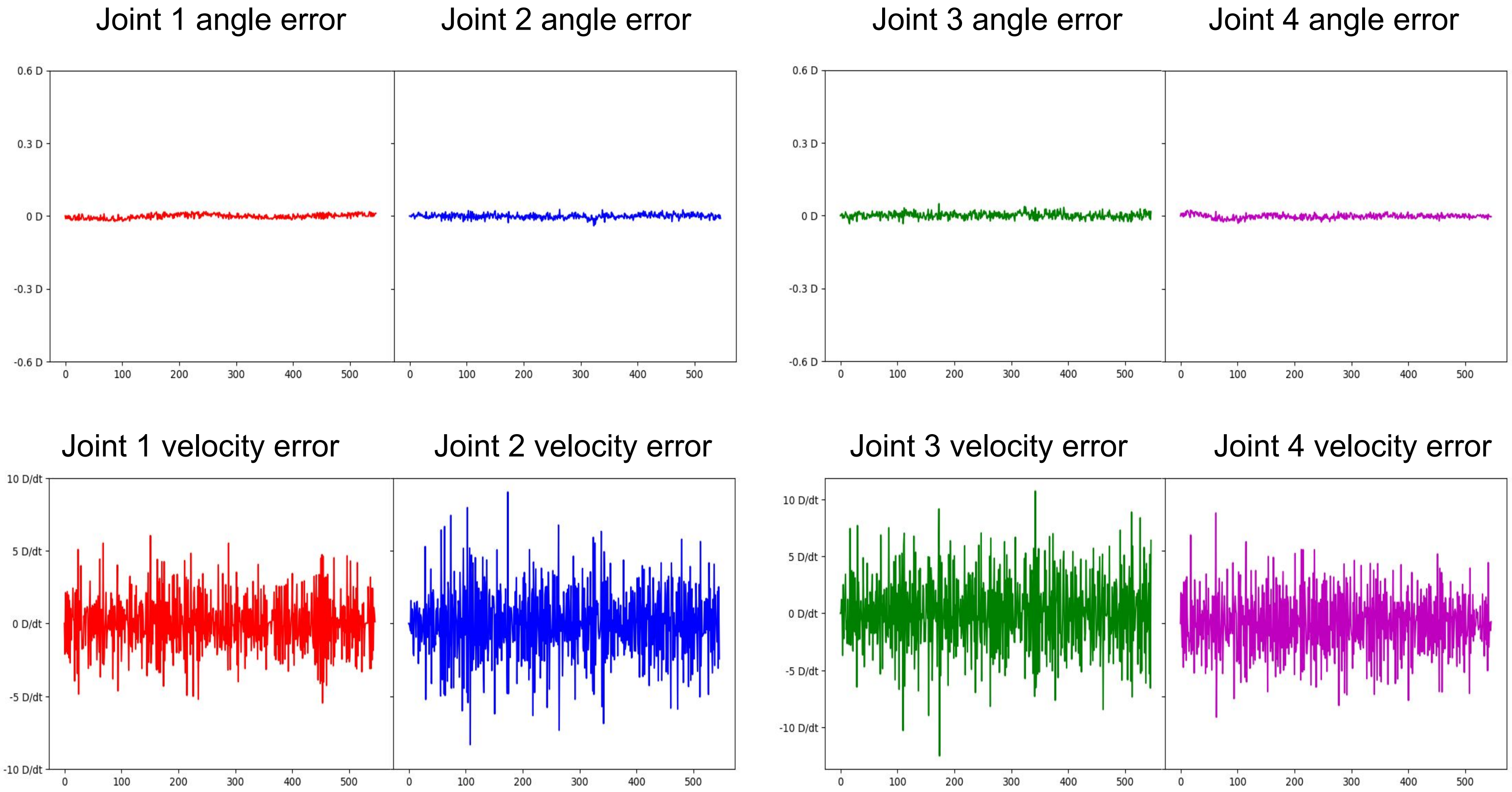
xy-plane xz-plane



Ee-distance to target:
0.09685061498675455

Time to reach the target:
62.77110314369202s

Errors over iterations of example run



Joint 1 angle error

Joint 2 angle error

Joint 3 angle error

Joint 4 angle error

Joint 1 velocity error

Joint 2 velocity error

Joint 3 velocity error

Joint 4 velocity error

A - Robot Vision - Target Identification/Detection

Method:

- Find threshold image of false and valid target same as before using inRange
  - Do not need to do template matching as targets can't be overlapped
- Find the contours and convex hull
  - The one that has the largest difference in area will be the target
- Find centre of target using moments

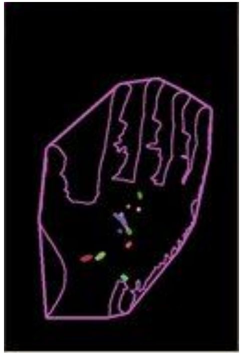
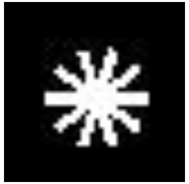


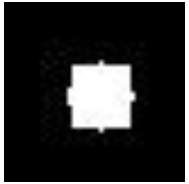
Image Source: OpenCV tutorial for convex hull

Results:

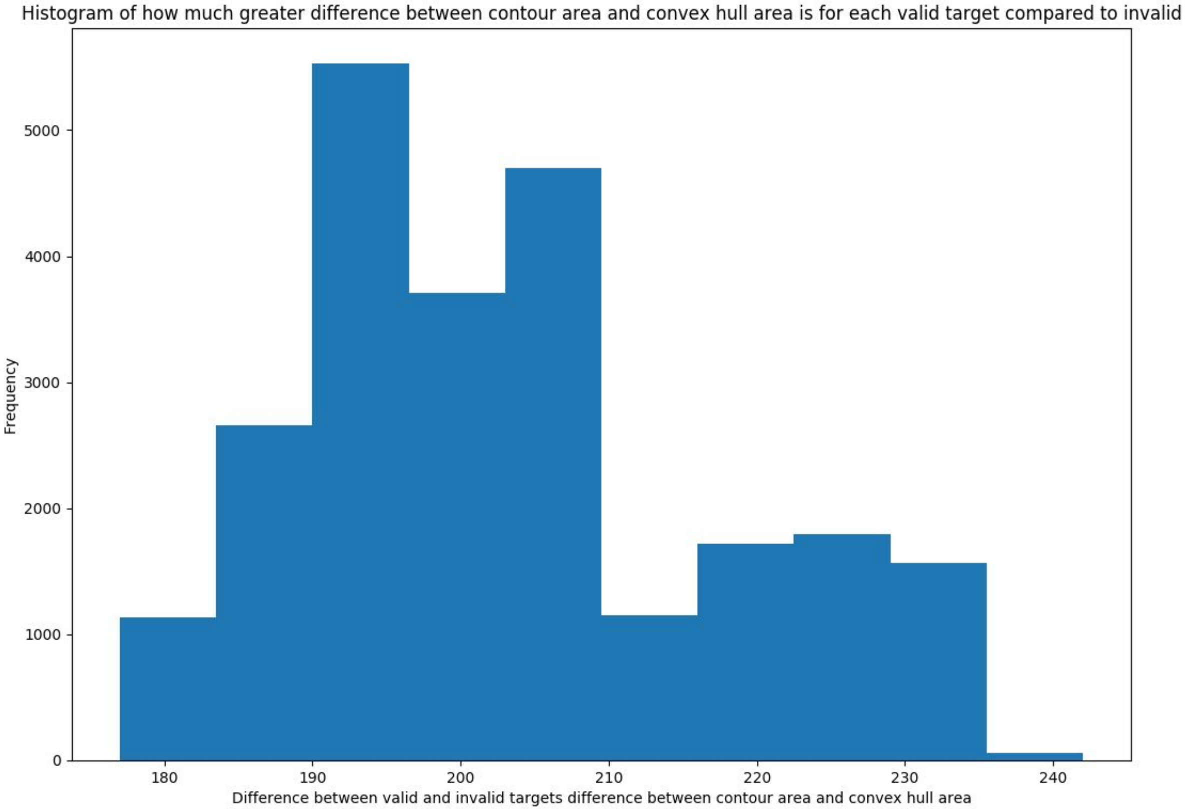
- Yet to find a case where valid target is mistaken for invalid target
- Valid and invalid target with smallest difference from each other were valid 72 and invalid 118, but they still had a difference of 177 (196 and 19 respectively)



Valid 72



Invalid 118

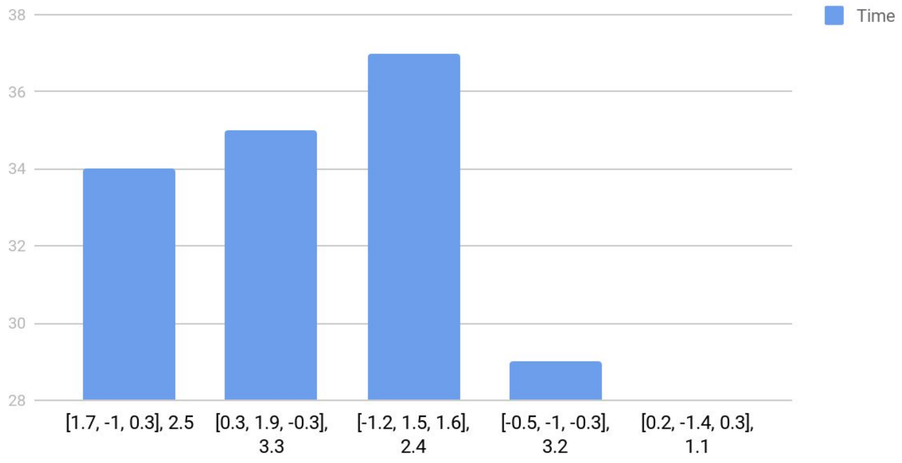


Methodology: Paired every valid image with every invalid image and found difference

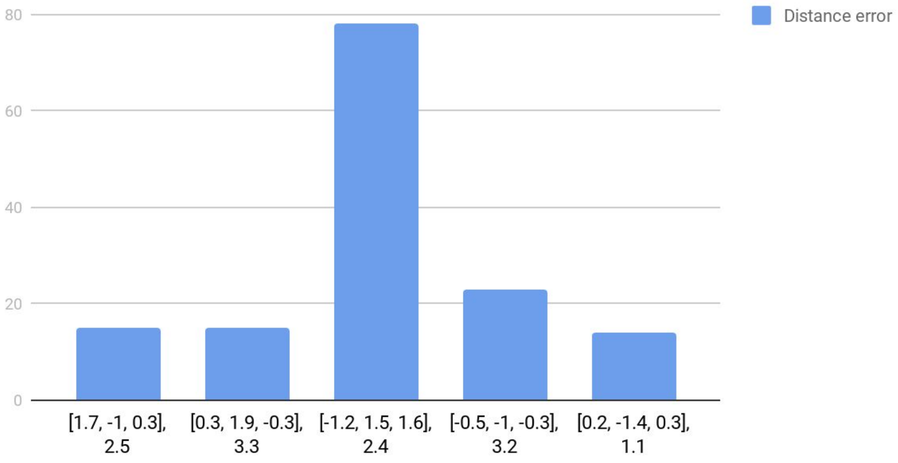
B - Robot Control - Inverse Kinematics Velocity Control

! The arm gets stuck in a local minimum approx. 1 in 4 times.

Average time needed to reach target in seconds 5x5



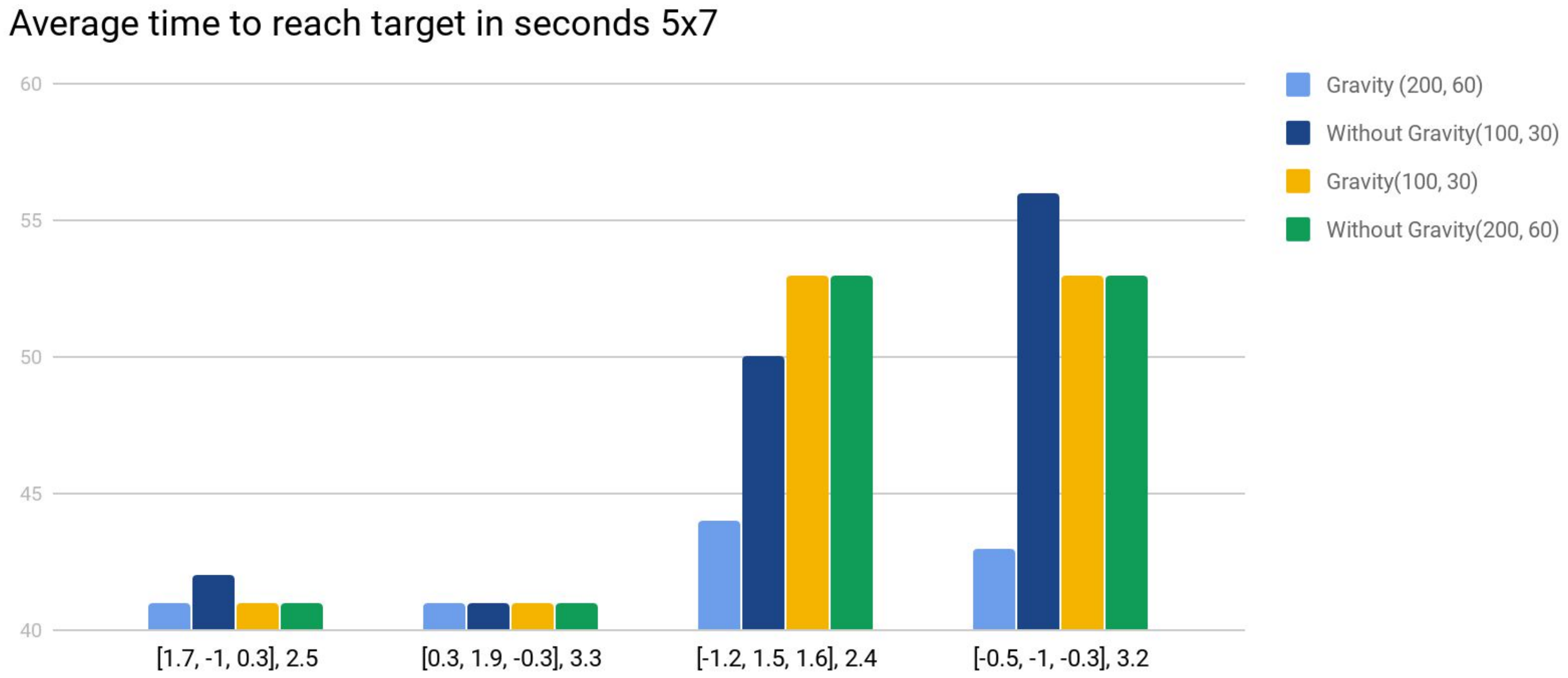
Average distance error Eepos to target xe3 in 5x5 runs



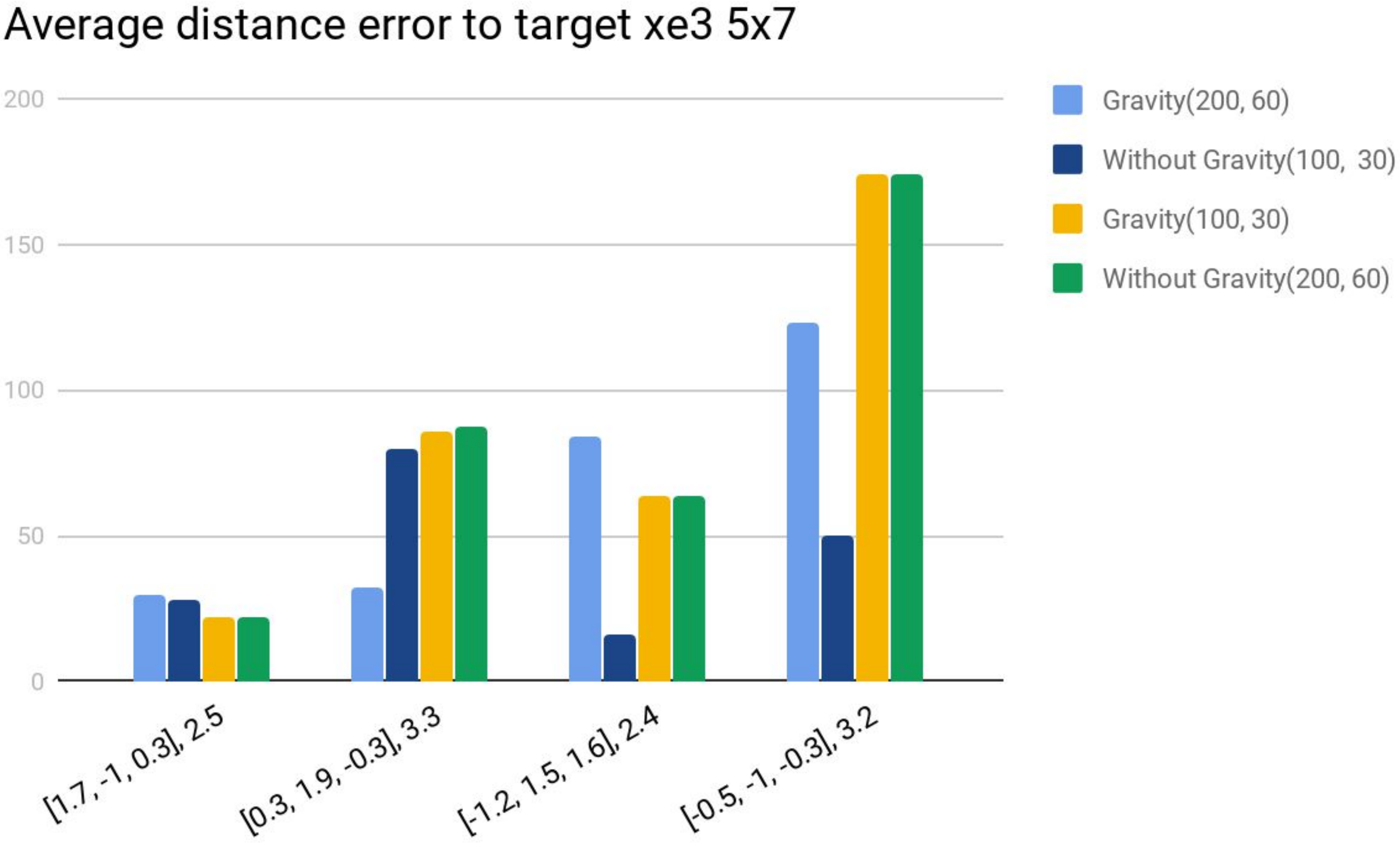
B - Robot Control - Gravity Compensated Torque Control

Out of 30 random runs it reached the valid target 30 times (Gravity/!Gravity)

Time needed for success for each point, Distance from target to end effector:



Distance error on success for each point, Distance from target to end effector:



P/D Effect Gravity on, TORQUE Target: [1.7, -1, 0.3], Distance to target: 2.5	P = 60	P = 120	P = 180
D = 20	Distance error: 0.0327 Time: 43.668s	Distance error: 0.0332 Time: 39.873s	Distance error: 0.04861 Time: 49.680s
D = 40	Moderately wiggly behaviour Distance error: 0.05510 Time: 63.367s	Wiggly behaviour Distance error: 0.02758 Time: 43.726	Distance error: 0.02389 Time: 35.841s
D = 60	Very wiggly behaviour Distance error: 0.08903 Time: 82.699s	Moderately wiggly behaviour Distance error: 0.04059 Time: 52.723s	Wiggly behaviour Distance error: 0.03239 Time: 42.584s

C - Open Challenge

Induced Force/dt:

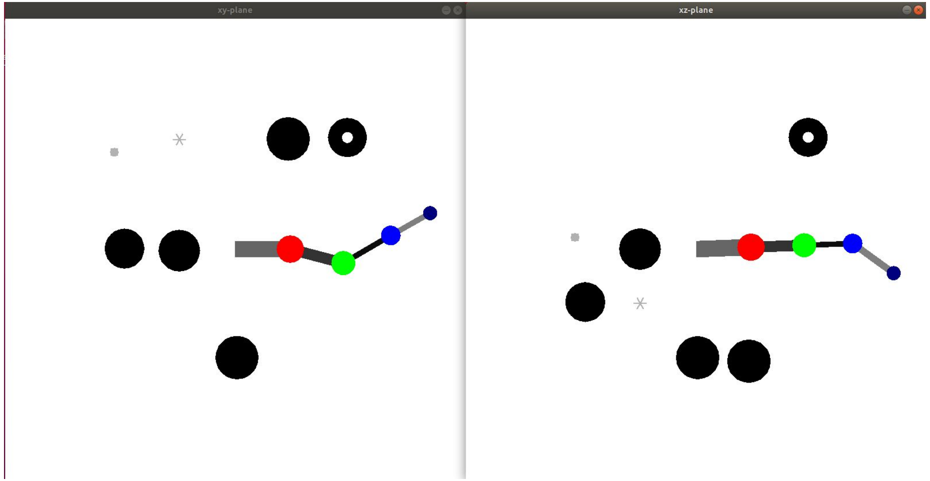
α constant for strength of force

β constant for radius of influence

$$\boldsymbol{\varphi}(x_1,x_2,r)=\begin{cases}\begin{matrix}0\\0\,,\\0\end{matrix} & d(x_1,x_2)>\beta r\\ \alpha\left(\frac{1}{d(x_1,x_2)}-\frac{1}{\beta r}\right)\left(\frac{1}{d(x_1,x_2)^2}\right)\left(\frac{x_1-x_2}{d(x_1,x_2)}\right), & d(x_1,x_2)\le\beta r\end{cases}$$

Screenshot:

white dot indicates closest obstacle


xy-plane
xz-plane