Introduction to Robot Operating System (ROS)

Application to mobile robots

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, May 31, 2019

BRADLEY University

Outline

- Introduction
  - Historical Background
  - Robot Programming Before ROS
  - ROS is ..
  - ROS Equation
  - Applications
- ROS Concepts
  - Filesystem
  - Computation Graph
  - Community level
- ROS installation
- Future of ROS

BRADLEY University

Intrduction

History and Legacy

- Started in 2007 by researches from Stanford AI Robot (Stair) and the Personal Robots (PR) Program and was sponsored by Willow Garage a visionary robotics incubator.
- Used Worlwide in Research and Industry.
- Currently supported by the Open Source Robotics Foundation.

Figure: Stair

- No common platform for develeoping robotics
- Build every thing from scratch
- Algorithm implementation

Intrduction

ROS is ..

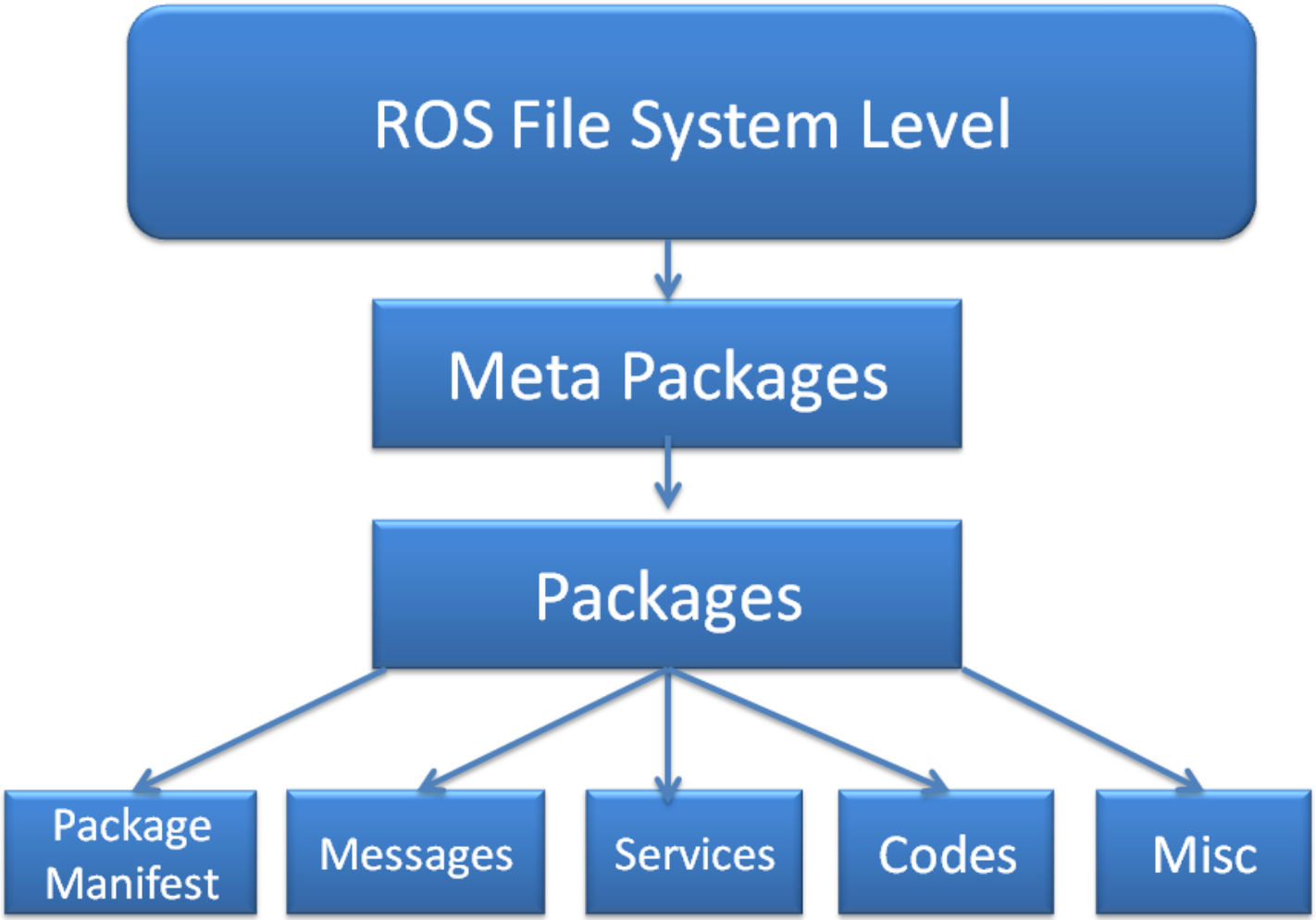
A flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.

Intrduction

Ros Equation

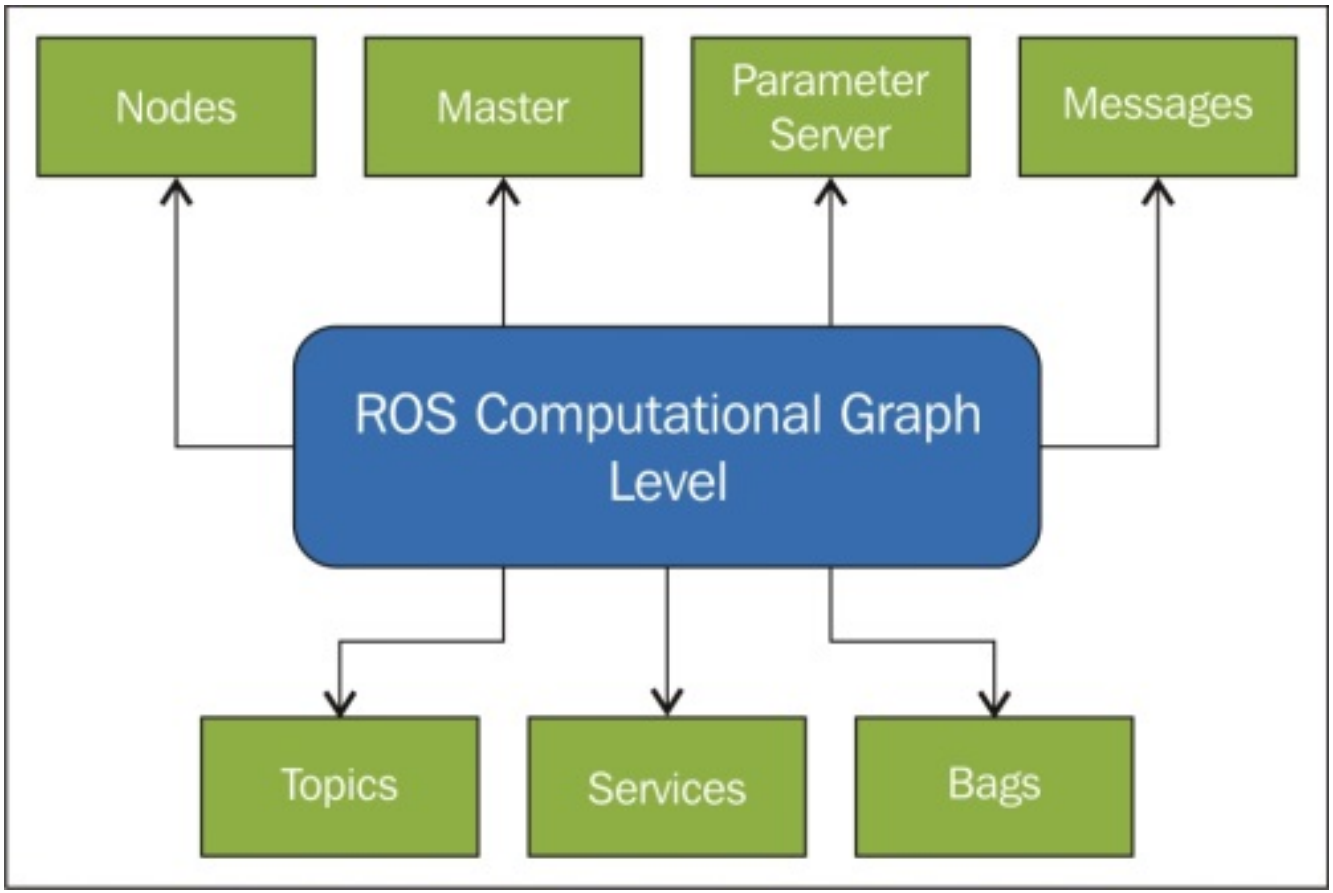
Intrduction

Applications


ROS File System Level
Meta Packages
Packages
Package Manifest
Messages
Services
Codes
Misc

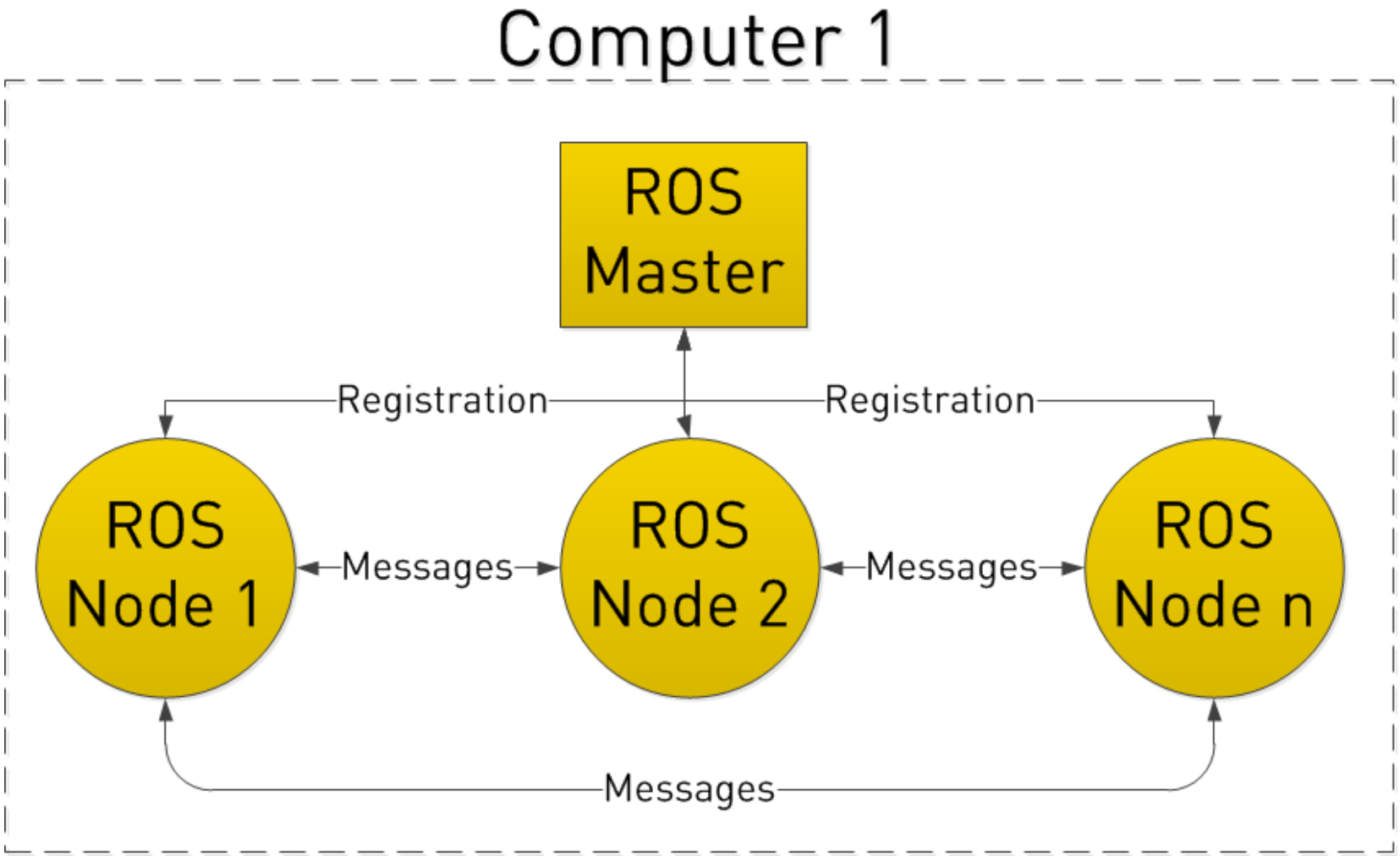
ROS Concepts

Computation Graph



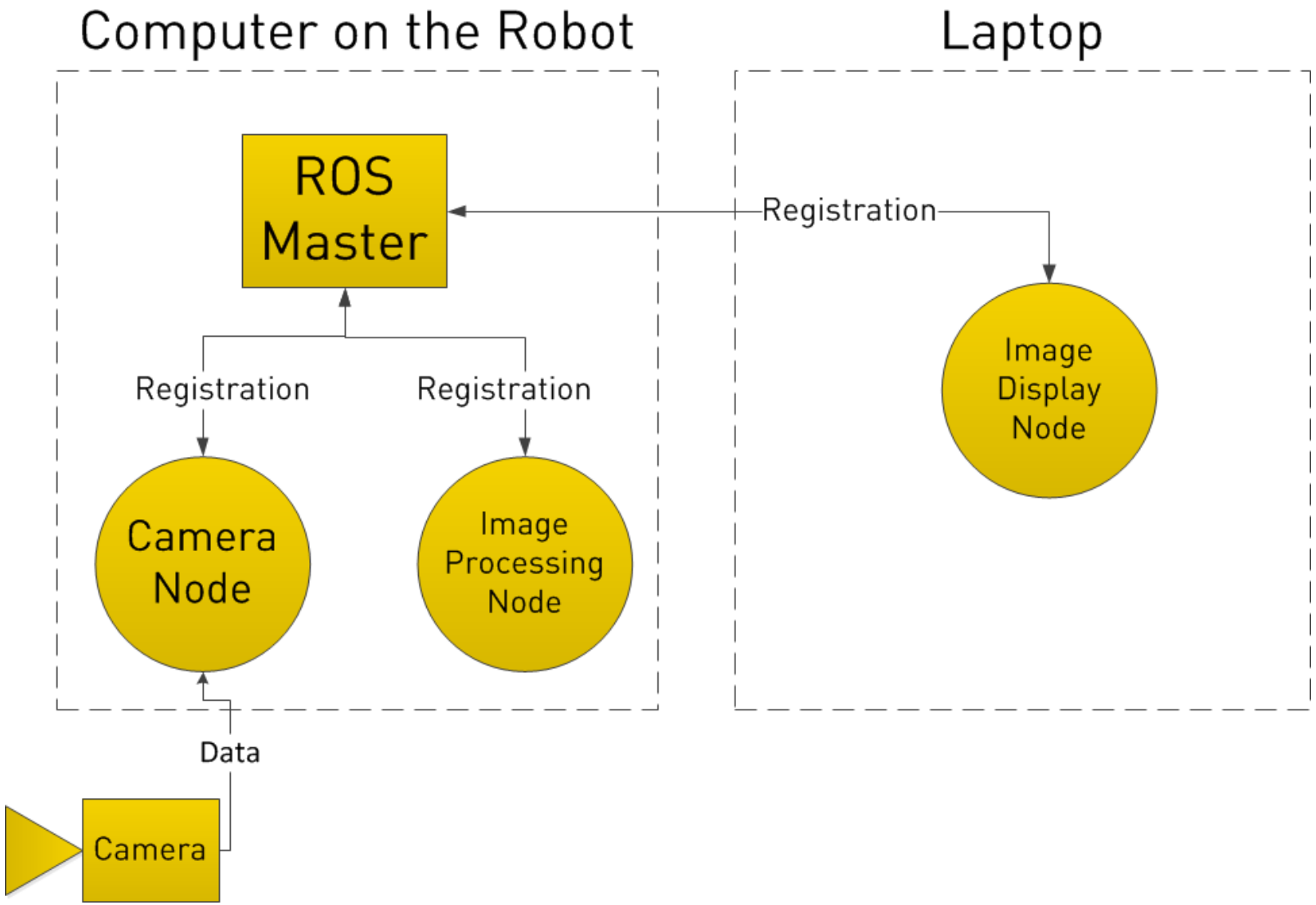
ROS Concepts

Computation Graph: Master



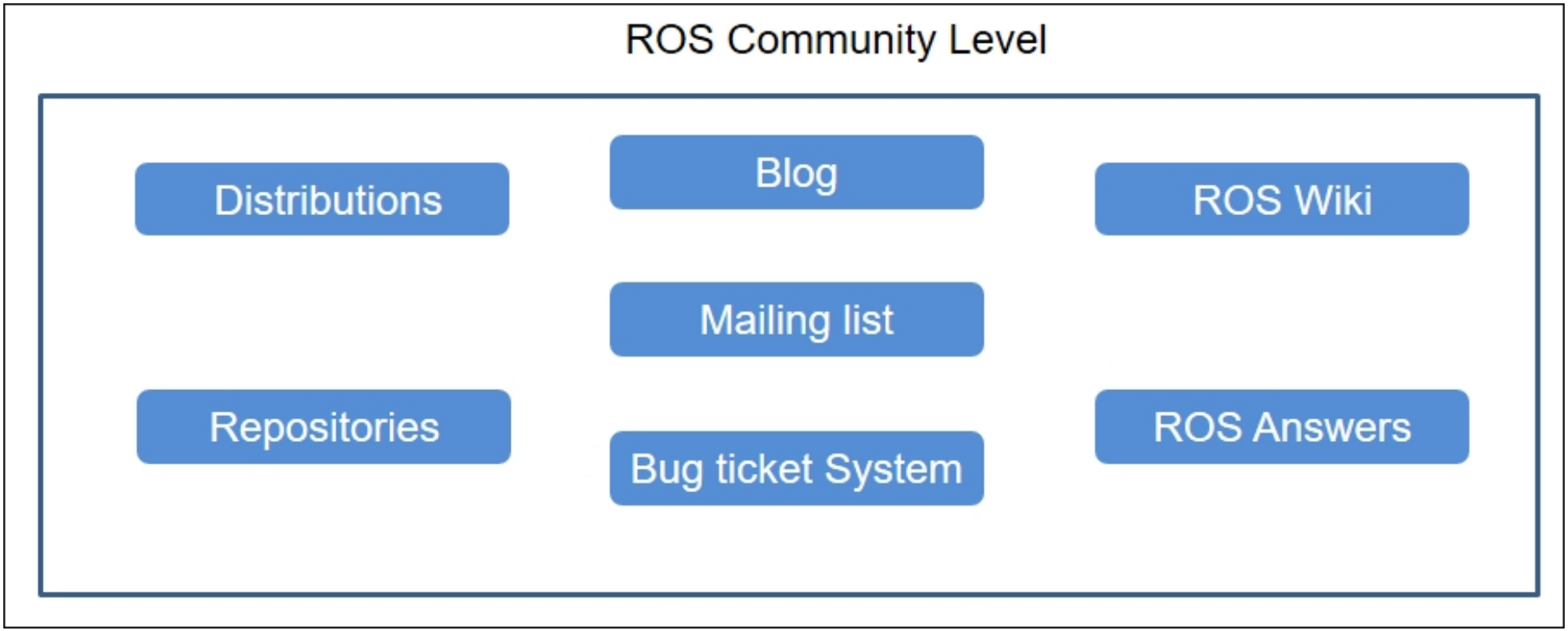
ROS Concepts

Computation Graph: Master



ROS Concepts

Community level



Installation

- Debian-based distributions such as Ubuntu.
- Many robots.
- Current supported distributions
  - ROS Kinetic Kame, Released May, 2016.
  - ROS Melodic Morenia, Released May, 2018

Installation

After choosing the distribution follow the instruction on ROS Wiki which start by:

- Configure your Ubuntu repositories.
- Setup your sources.list.
- Set keys.
- Install with "sudo apt-get install ros-kinetic-desktop-full".

Future of ROS

- Security
- Critical Missions
- Distributed Processing

Thanks !

Matlab Robotics Systems Toolbox

Application to mobile robots

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, June 4, 2019

BRADLEY University

Outline

- Introduction
- Workflow
  - Desktop prototyping
  - Standalone ROS Nodes
- Examples

Introduction

According to mathworks.com

Robotics System Toolbox provides algorithms and hardware connectivity for developing autonomous robotics applications for aerial and ground vehicles, manipulators, and humanoid robots. Toolbox algorithms include path planning and path following for differential drive robots, scan matching, obstacle avoidance, and state estimation. For manipulator robots, the system toolbox includes algorithms for inverse kinematics, kinematic constraints, and dynamics using a rigid body tree representation.

Workflow

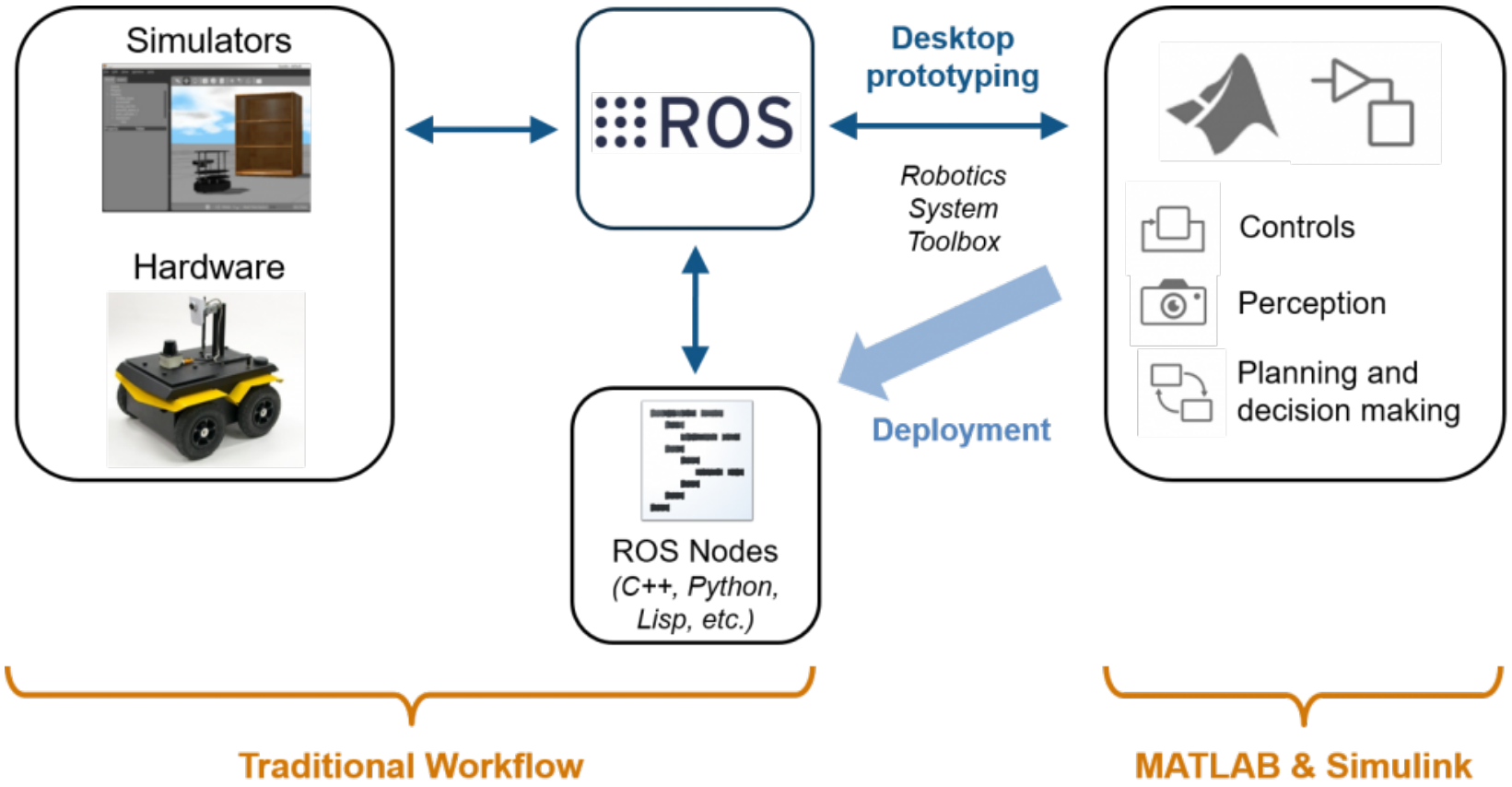


Figure: Matlab robotics tool box and ROS workflow. courtesy of mathworks.com

Workflow

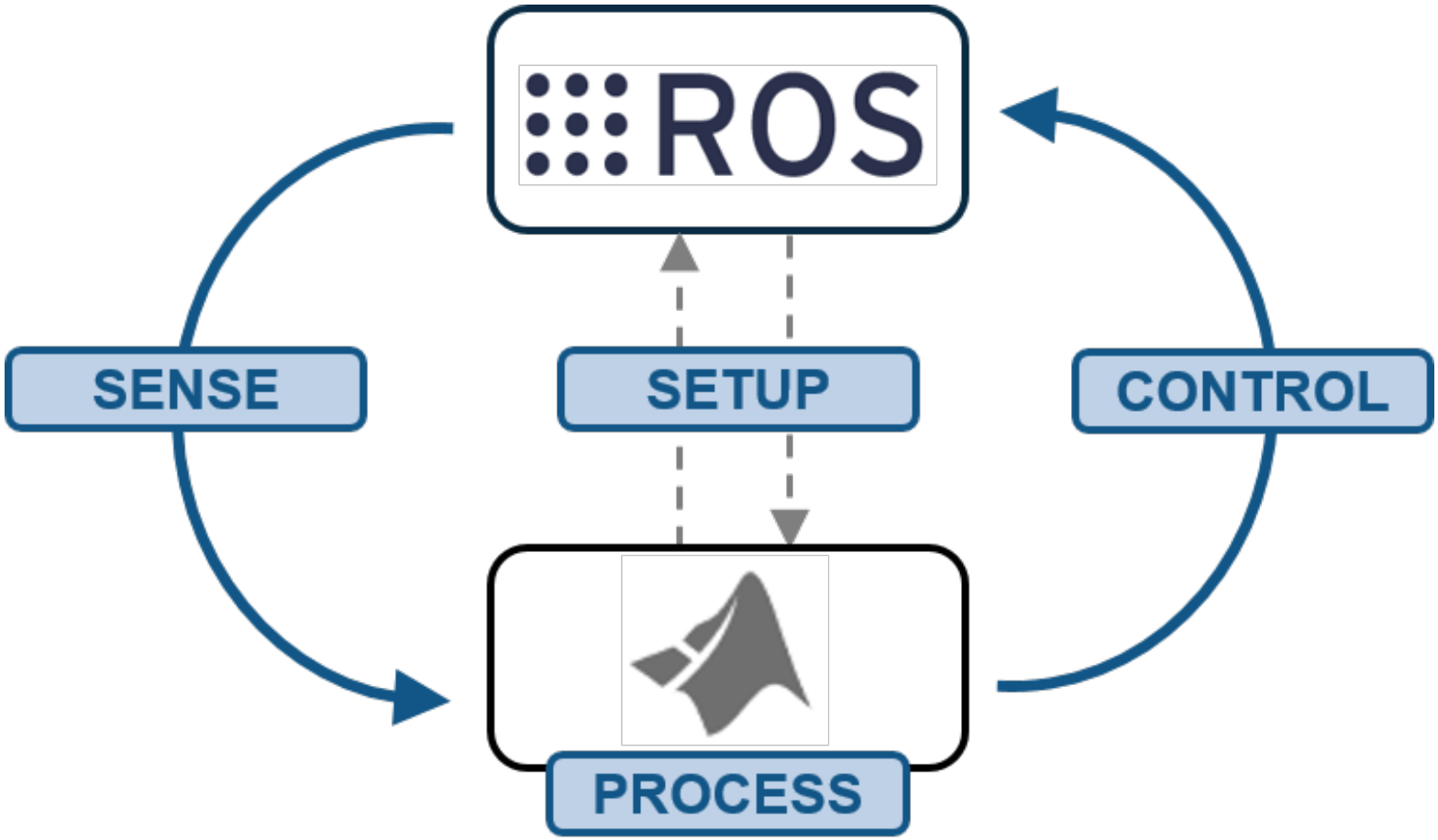


Figure: Matlab and ROS integration, courtsey of mathworks.com

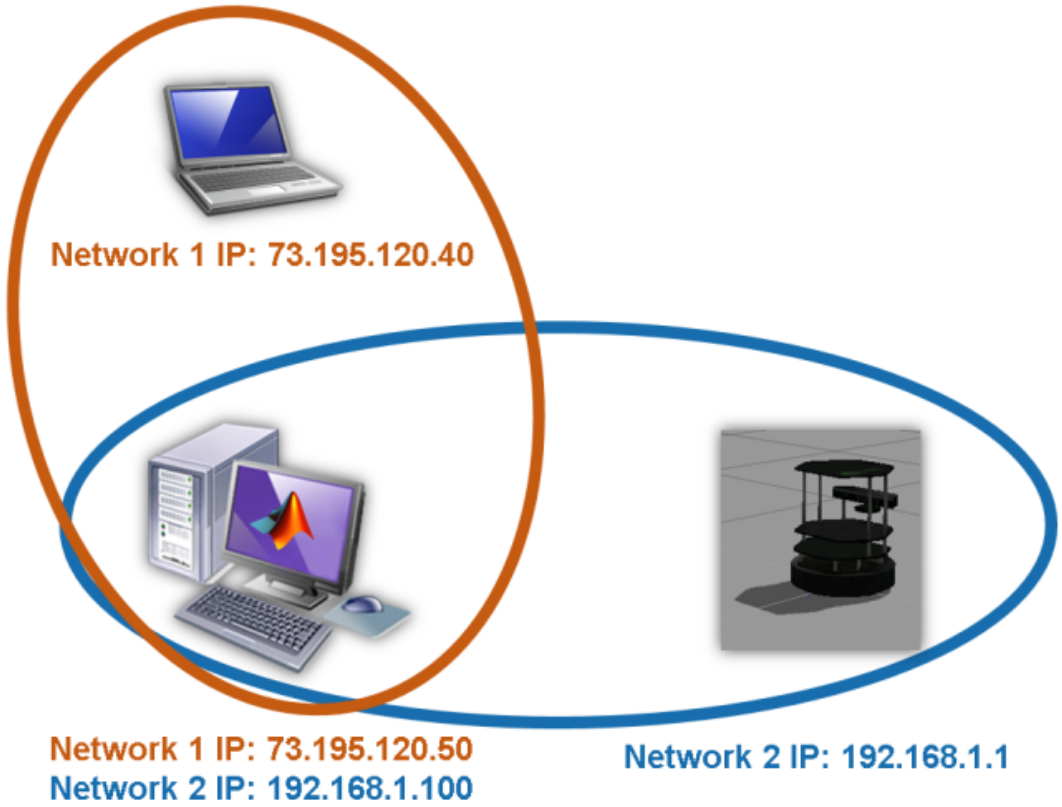
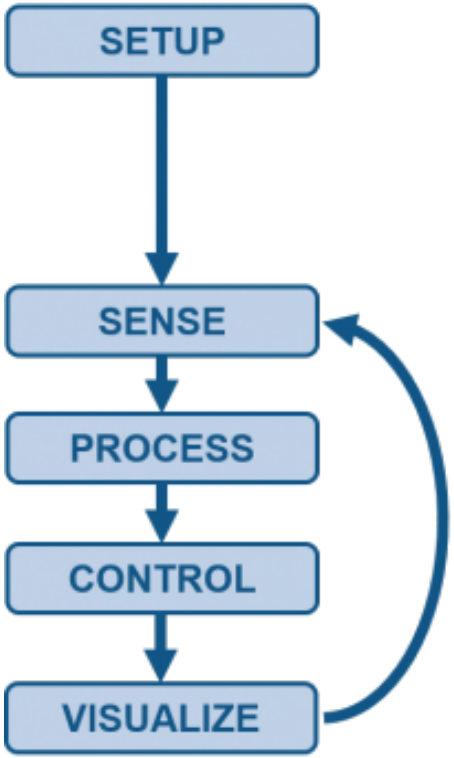


Figure: Matlab ROS desktop prototyping, mathworks.com

Workflow

Desktop prototyping




```
rosinit('ipAddress')
mySub = rossubscriber('/sub_topic');
[myPub,pubMsg] = rospublisher('/pub_topic');
currentTime = 0;

tic
while(currentTime < 10)
  recvMsg = mySub.LatestMessage;

  ctrlOut = myAlgorithm(recvMsg);

  pubMsg.FieldName = ctrlOut;
  send(myPub,pubMsg);

  currentTime = toc;
  plot(currentTime,ctrlOut)
end
```

Figure: Desktop prototyping code template, courtsey of mathworks.com

Worflow

Standalone Node

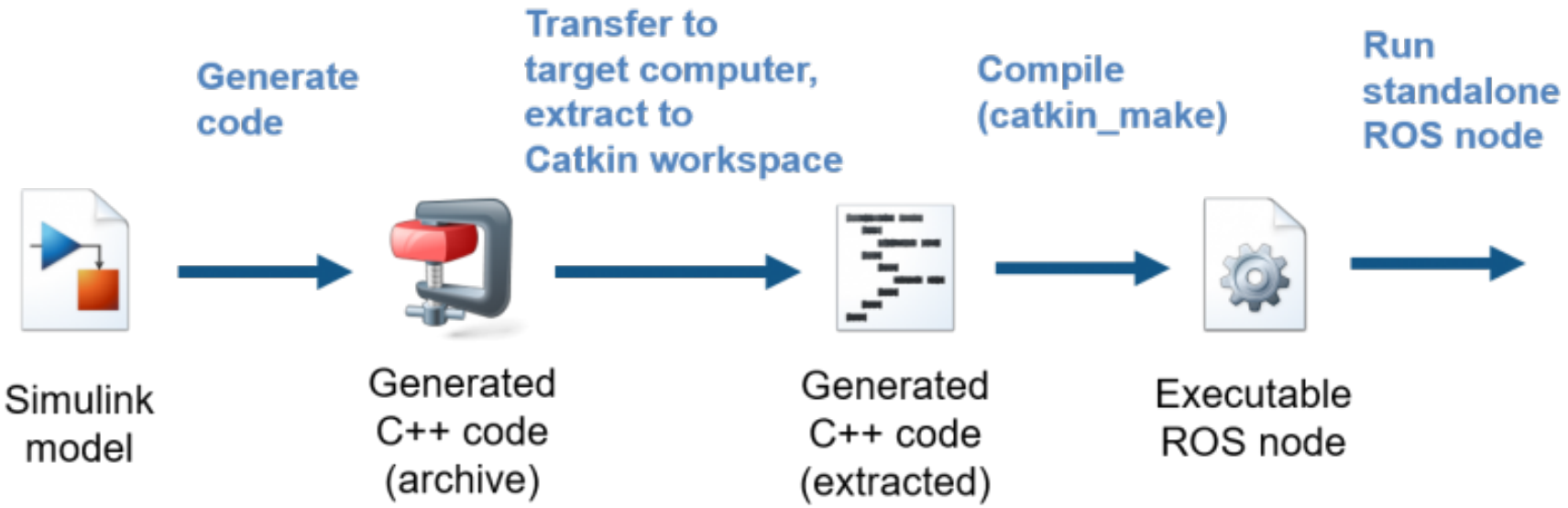


Figure: Generation of ROS standalone node, courtsey of mathworks.com

Workflow

Standalone Node

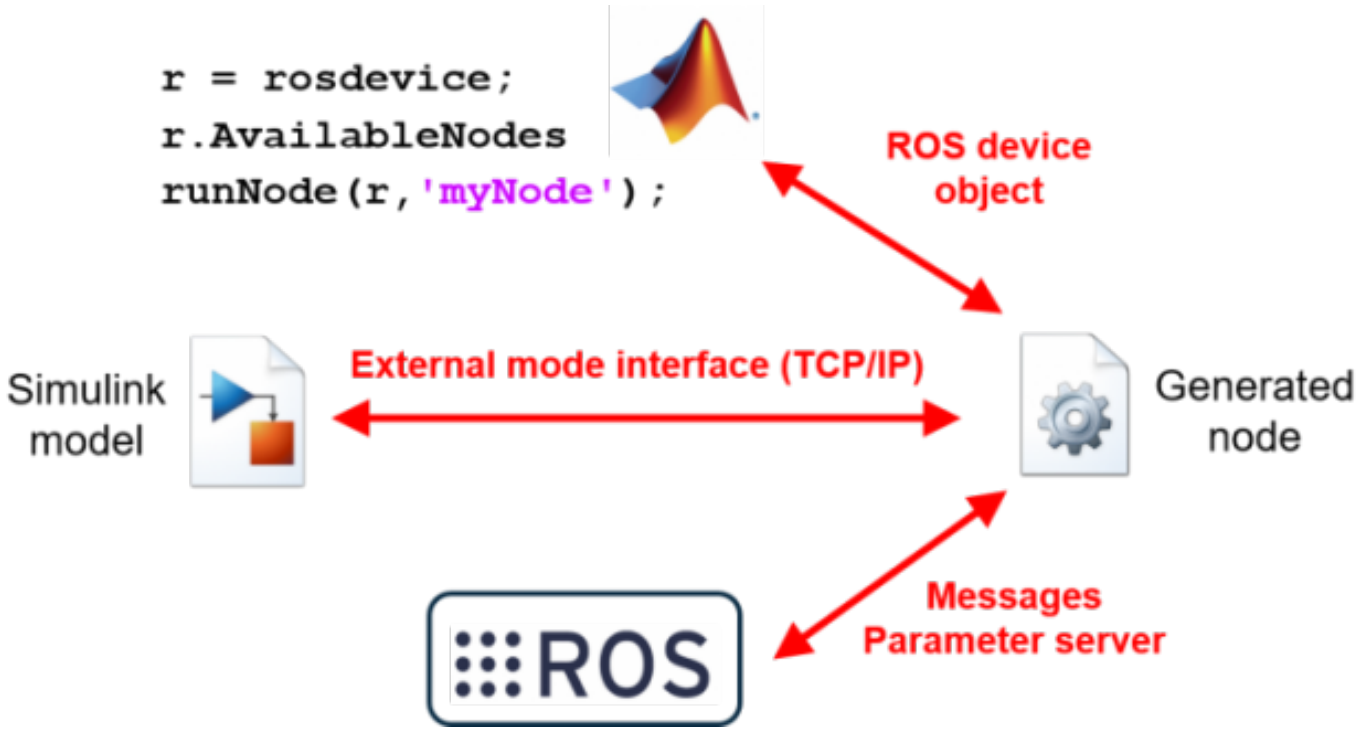


Figure: Access to ROS standalone node, courtsey of mathworks.com

Examples

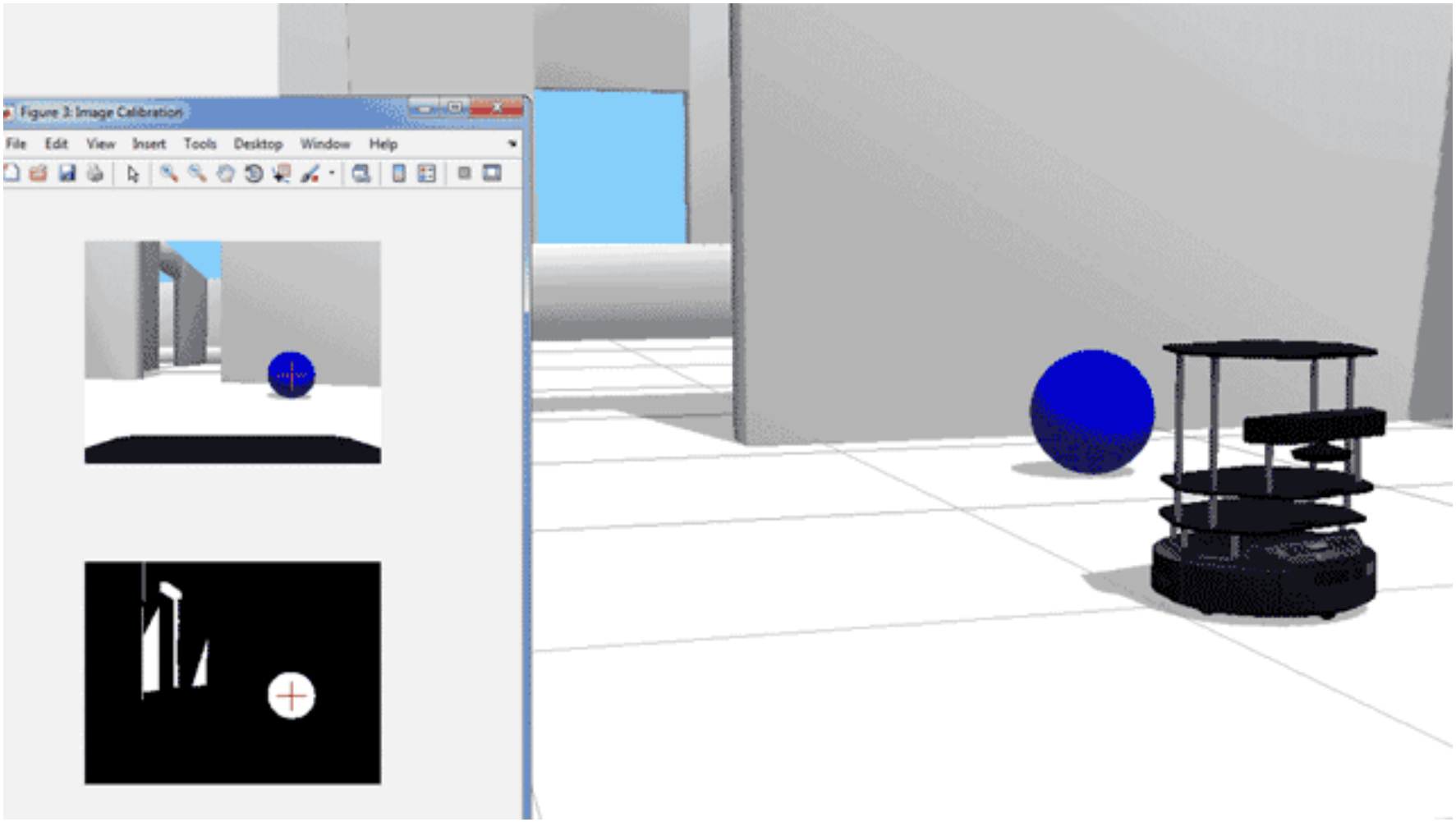


Figure: Turtle bot example, courtsey of mathworks.com

Area Coverage Optimization

Progress Report

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, June 21, 2019

BRADLEY University

Outline

- Introduction to V-REP
- Interfacing Matlab and ROS on the same Machine
- Line following simulation
- Leader follower simulation
- Area Coverage simulation
- Future Work

V-REP

General purpose robot simulator with integrated development environment "coppeliarobotics.com".

Interfacing

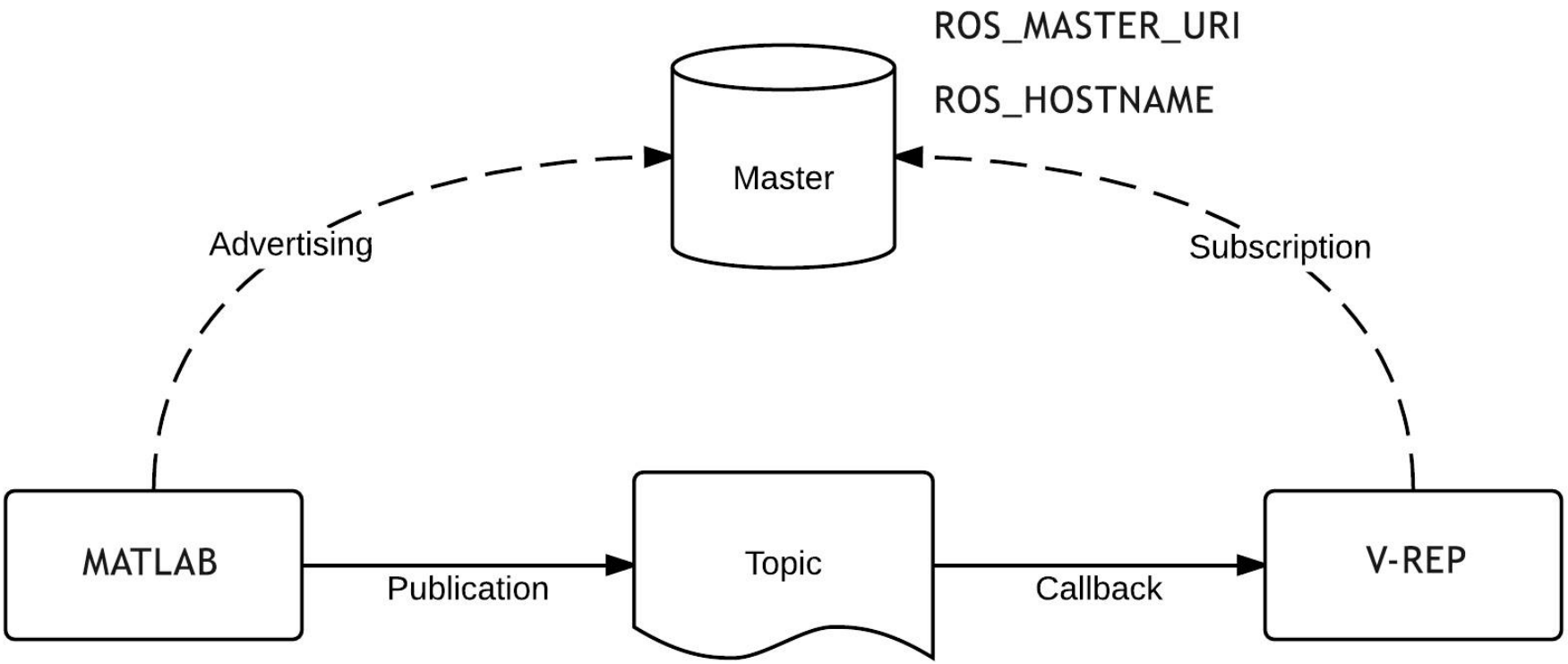


Figure: ROS, Matlab and V-REP interface

Line Following

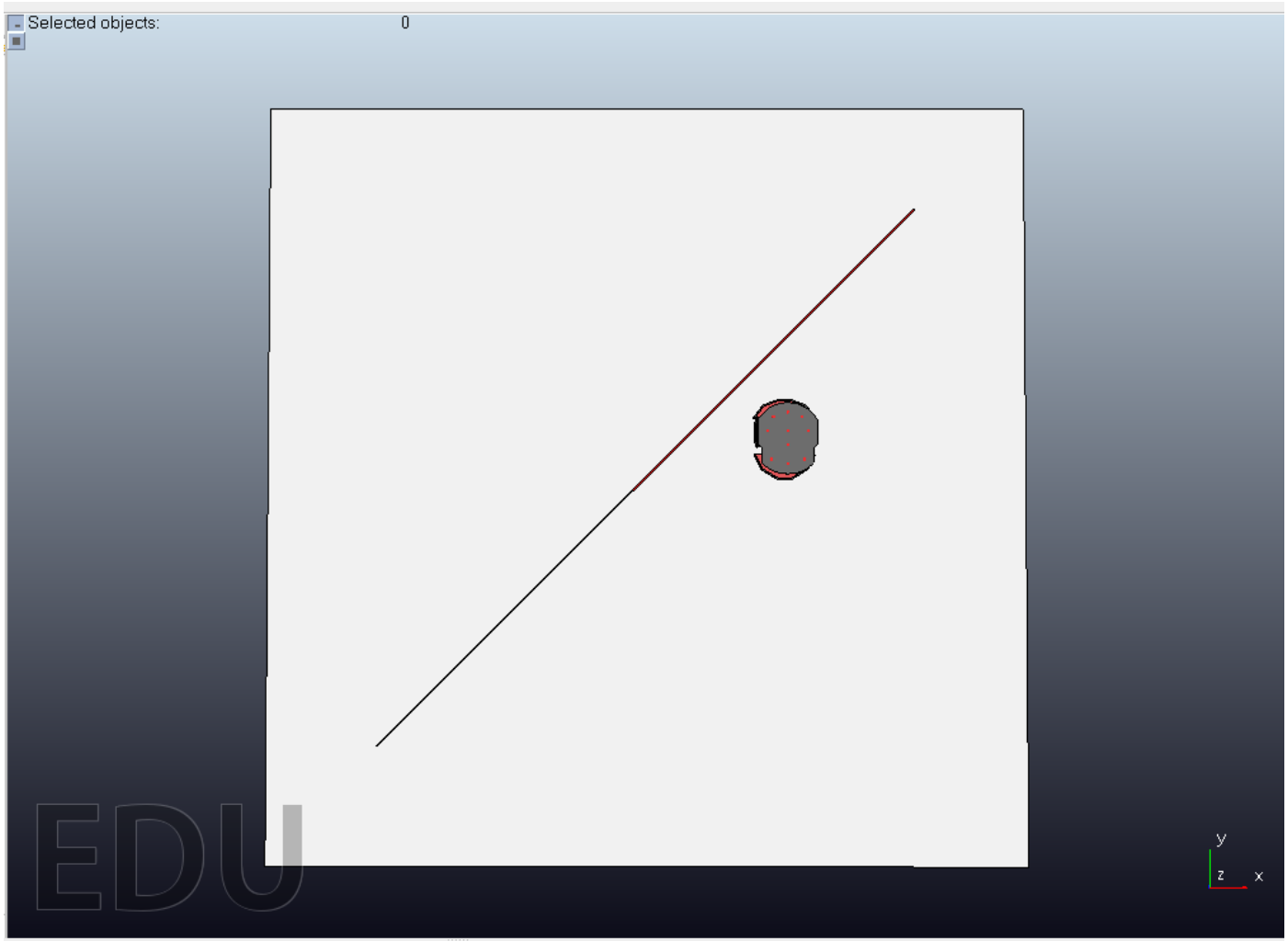


Figure: Line Following Scene

Leader Following

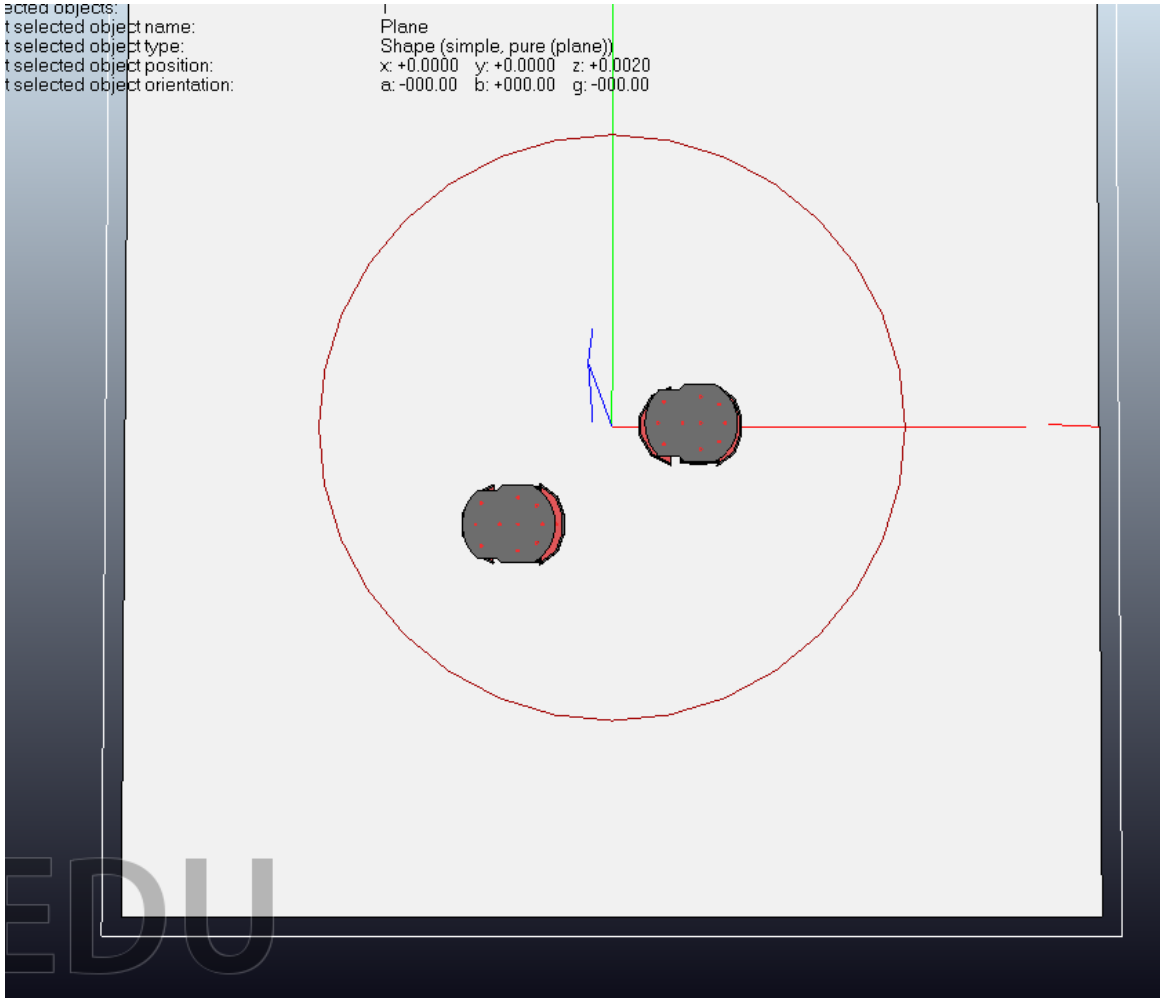


Figure: Leader Follower Scene

Area Coverage

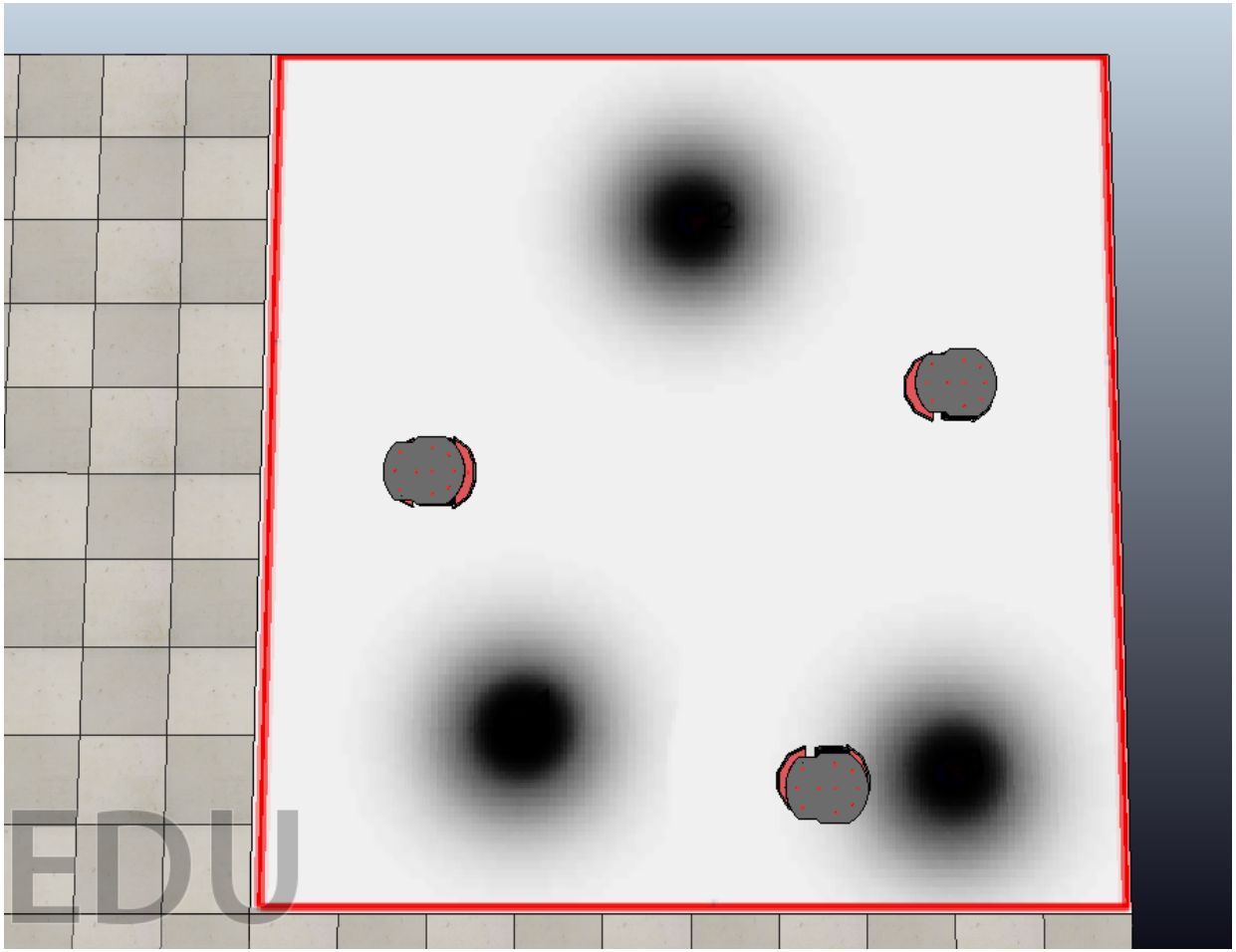


Figure: Area Coverage Scence

Future Work

- Expiremental Validation.
- Refining simulation results.

Questions?

Area Coverage Optimization

Progress Report

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, July 5, 2019

BRADLEY University

Outline

- Objectives
- Refining Simulation

Objectives

- Refining Simulation
- Expiremental Validation

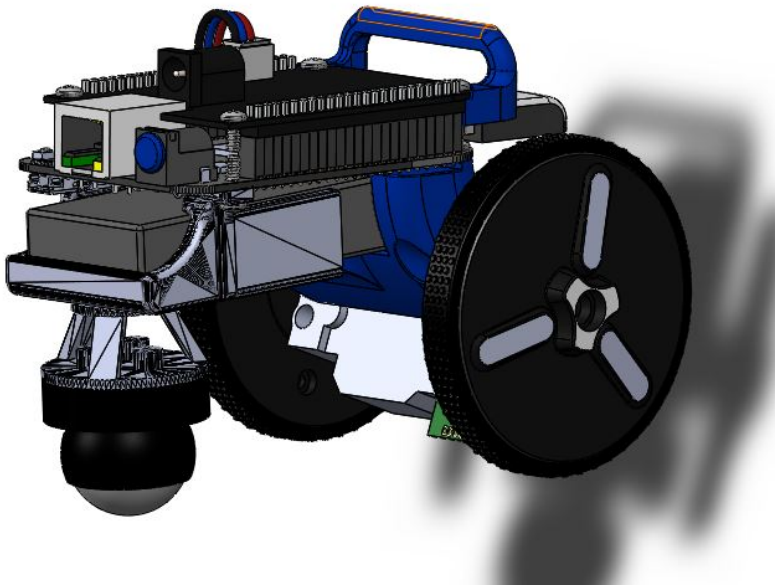


Figure: eduMOD Solidworks model

Refining Simulation

importing to V-rep

- Universal Robotic Description Format
- From Solidworks to URDF

Questions?

Area Coverage Optimization

Progress Report

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, July 19, 2019

BRADLEY University

Outline

- V-rep simulation
- Implementation

Solidworks to URDF plugin

- Tested with simpler models but kept getting the same error

```
rospack find Assem1
rospack failed: Command 'rospack find Assem1' exited with
status 1.
```

- the error is related to rospack find
- testing the urdf file with gazebo and rviz along with windows version of vrep

Implementation

- successfully interfacing matlab robotics toolbox with the eduMOD robot through cable and wifi.
- implemented the line following and leader follower trials and waiting for the recent version of area coverage code to be implemented.
- looking deeply into results.

Questions?

Area Coverage Optimization

Progress Report

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, August 02, 2019

BRADLEY University

Outline

- Milestones
- Refining simulation
- Implementation

Milestones

- Understand ROS, Matlab robotics Tool box, Vrep,and their interfacing
- Run the simulation demos using pioneer robot and then refining the simulation to get better results
- Understand how to navigate beaglboneblue through ssh
- Implement the area coverage algorithm with eduMIP robot

eduMIP urdf

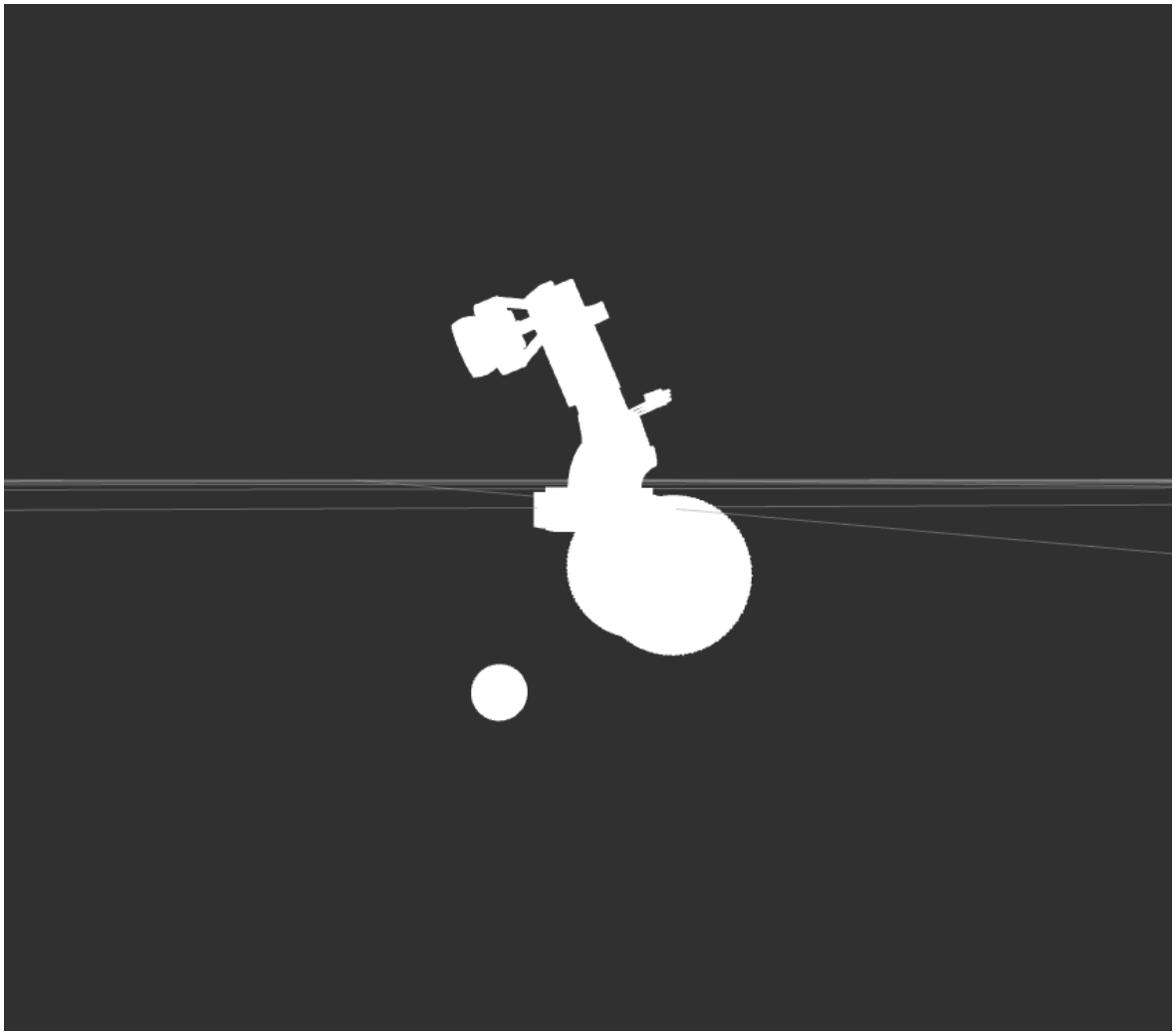


Figure: eduMIP rviz

Implementation

line following

Implementation

leader follower

Implementation

area coverage

- error in orientation calculation.

Area Coverage Optimization

Progress Report

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, August 30, 2019

BRADLEY University

Outline

- eduMip and eduMod
- Area Coverage optimization algorithm
- Deep reinforcement learning

EduMip

- low cost open platform for learning purposes developed by university of california San Diego.
- Comatible with ROS, Python, Simulink and Labview.

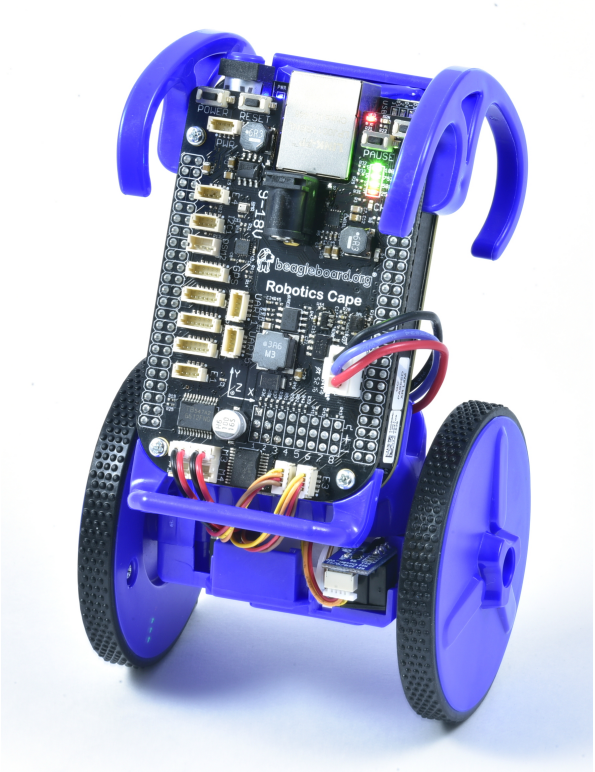


Figure: Edu Mip

EduMod

- Edu Mip was customized at Bradley university to act as a three wheeled robot instead of the inverse pendulum configuration to achieve more stability.

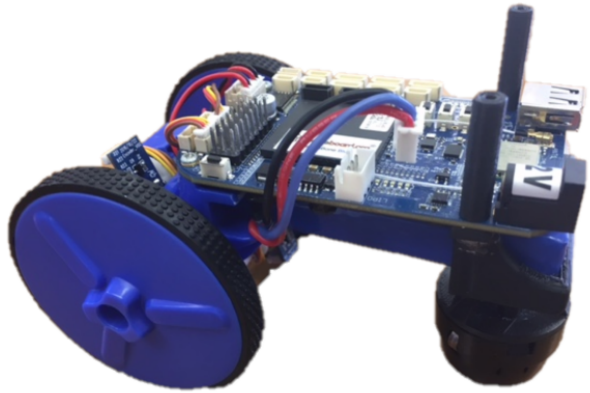


Figure: Edu Mod

Area Coverage Algorithm

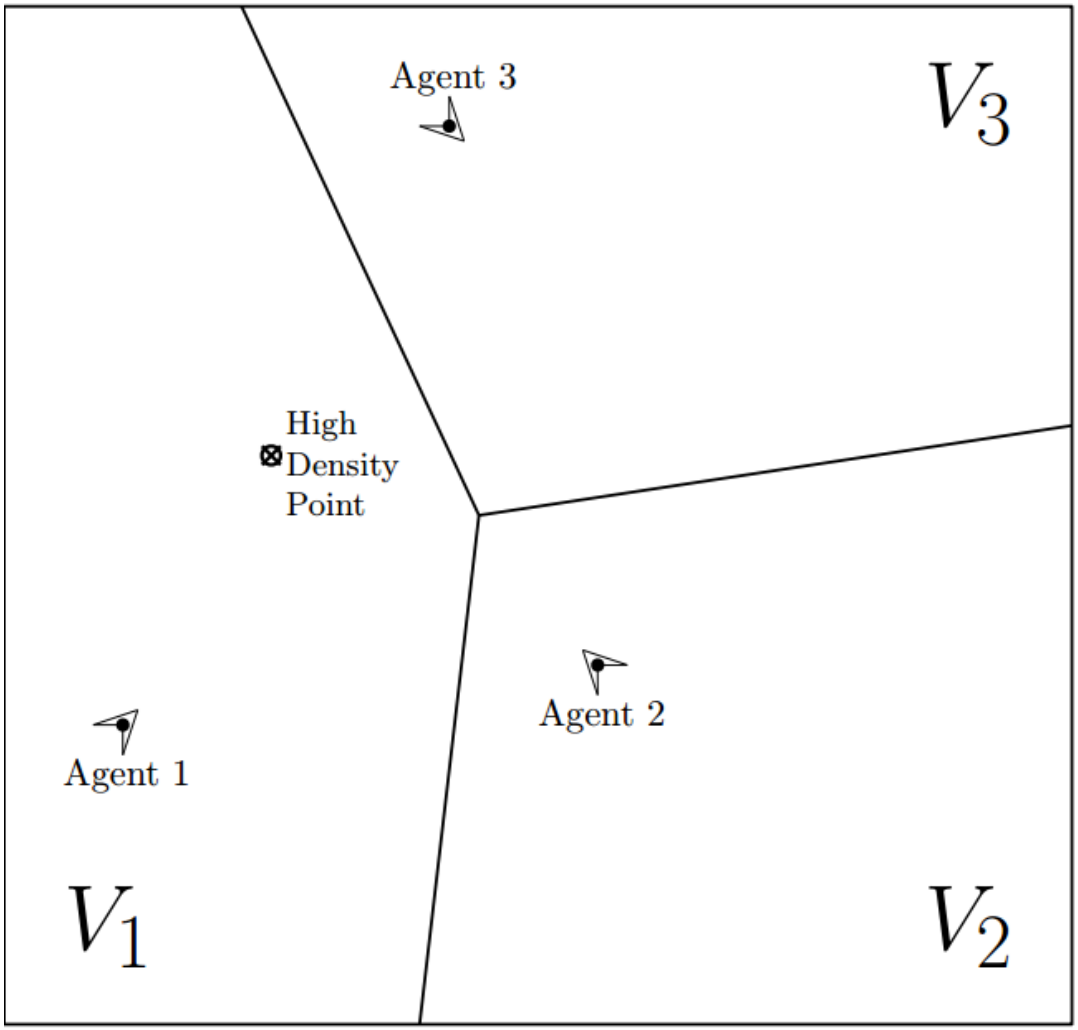


Figure: Voroni regions

Area Coverage Algorithm

Implementation

Deep Reinfrocement Learning

According to Wikipedia

Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward

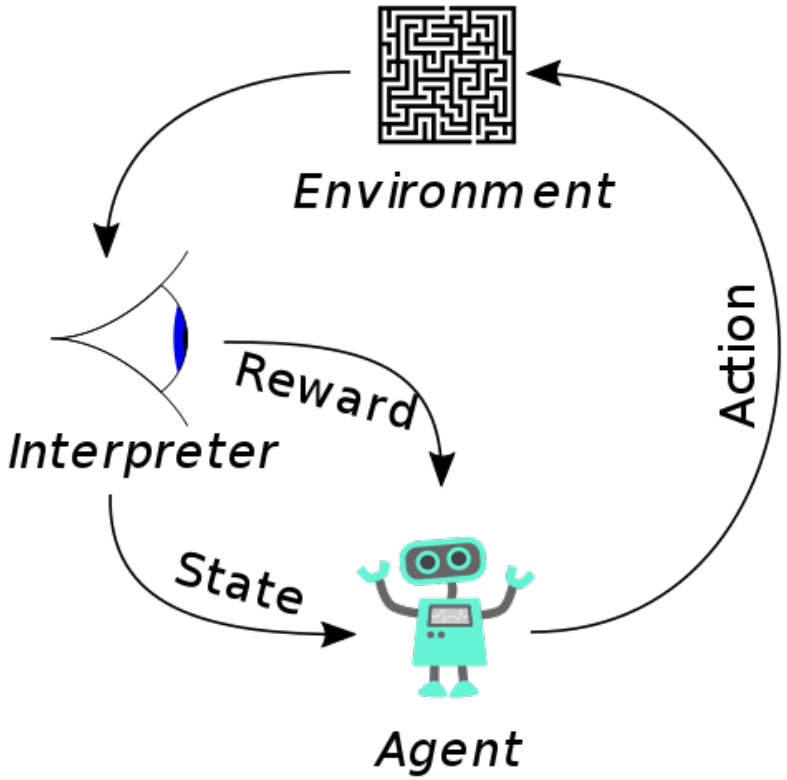


Figure: Typical framing of reinforcement learning, "wikipedia"

Deep Reinfocement Learning

Enviroment

A physical space that robots with each other on.

Agents

The two robots, leader and follower.

Action

Robots movement

Reward

follower robot will obtain if error is minimized.

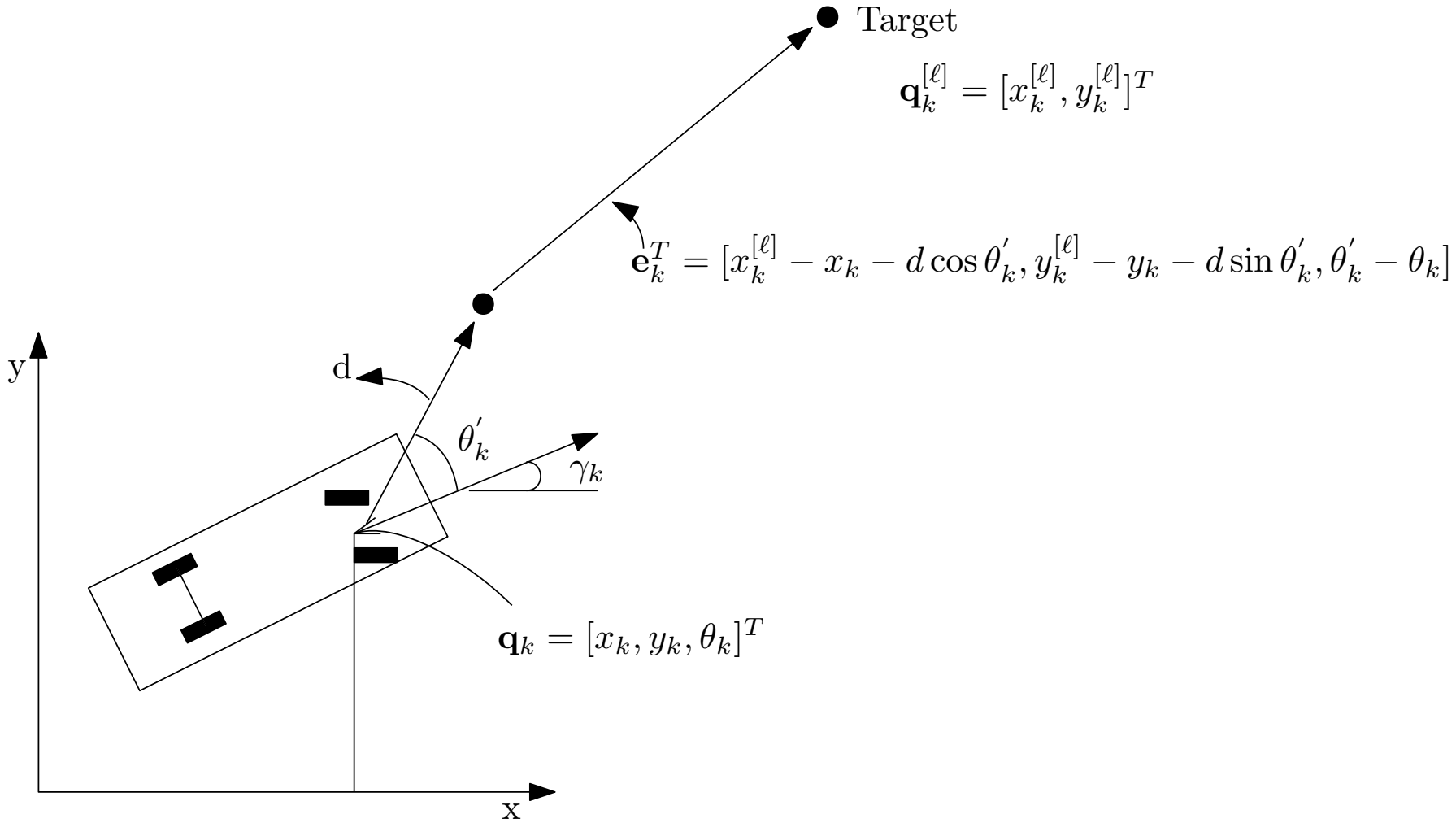


Figure: Leader Follower Problem setup

Questions?

Area Coverage Optimization

Progress Report

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, September 13, 2019

BRADLEY University

Outline

- Problem Setup
- Model Free Reinforcement Learning Algorithm
- Results
- Future work

Problem setup

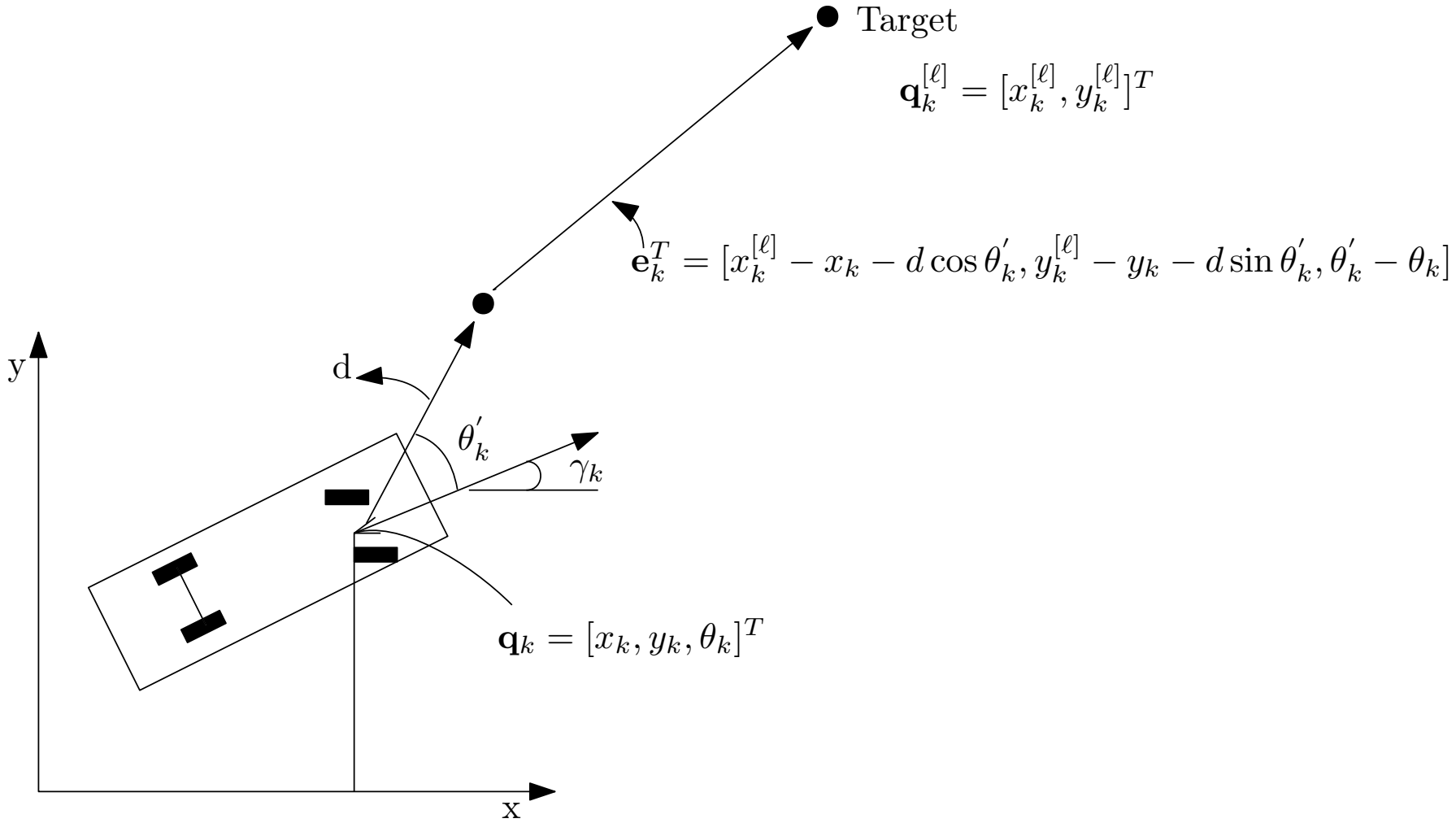


Figure: Leader Follower Problem setup

Important equations

$$\mathbf{e}_k = \left[x_k^{[\ell]} - x_k - d\cos\theta_k', y_k^{[\ell]} - y_k - d\sin\theta_k', \theta_k' - \theta_k \right] \tag{1}$$

$$\mathbf{u}_k^* = -\mathbf{P}_{uu}^{-1}\mathbf{P}_{ue} * \mathbf{e}_k \tag{2}$$

Results

- the wheights start to diverge after a short period of time.
- wheight matrices are not stable.

Future Work

- Iterating different initial conditions.
- adding sections in the matlab code to prevent the wheights from diverging.
- verify matrices stability.

Questions?

Area Coverage Optimization

Progress Report

Amr Elhussein
Advisor: Dr. Suruz Miah

Department of Electrical and Computer Engineering
Bradley University
1501 W. Bradley Avenue
Peoria, IL, 61625, USA

Friday, September 27, 2019

BRADLEY University

Outline

- Problem Setup
- Model Free RL Algorithm
- Results

Problem setup

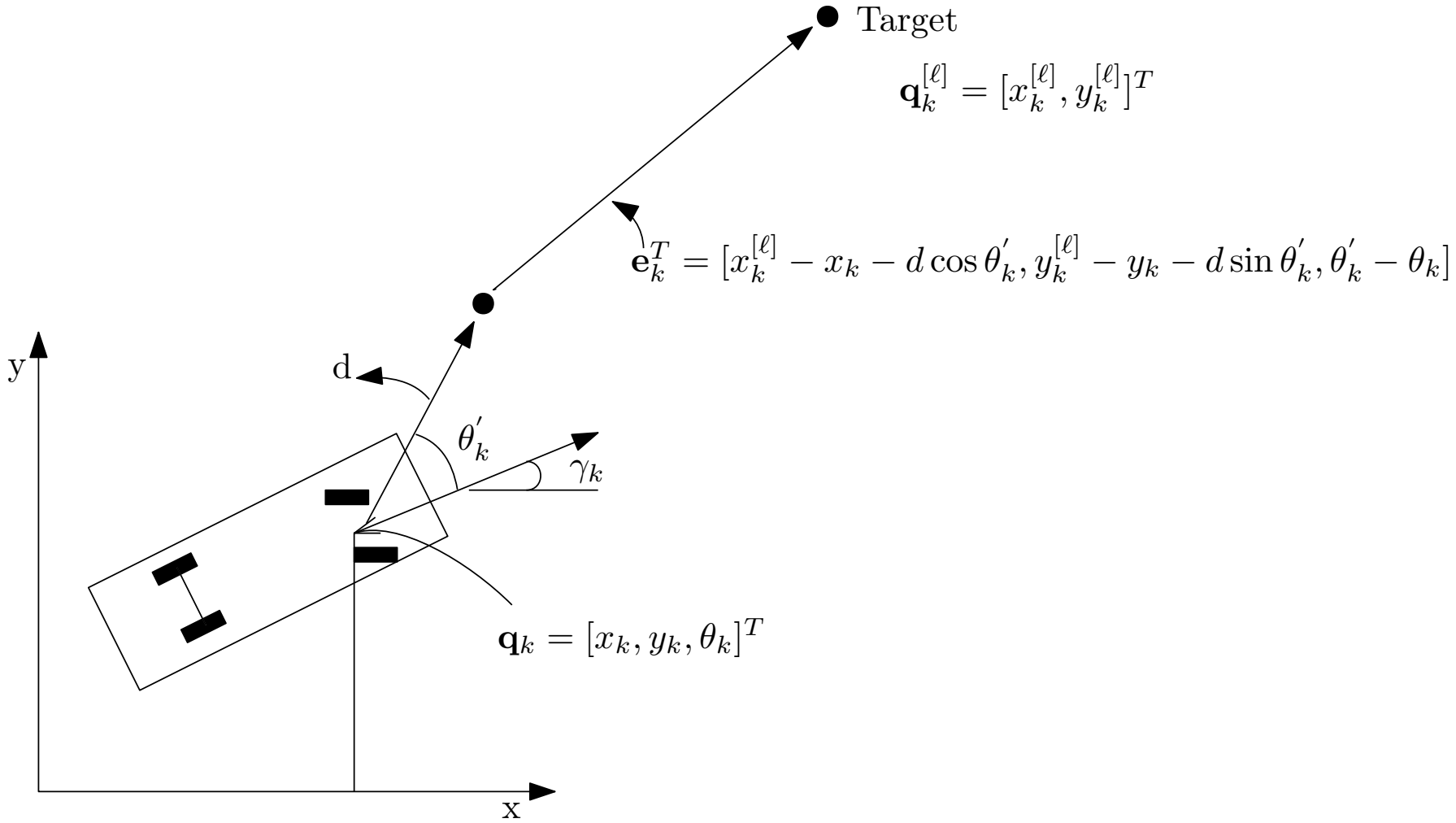


Figure: Leader Follower Problem setup

Model Free RL Algorithm

Algorithm 1: Model-free leader-follower.

Input: Sampling time T, $\mathbf{e}_0$, $\mathbf{Q}$, $\mathbf{R}$, and threshold ε
Output: Optimal error tracjectory $\mathbf{e}_k$, for $k = 0, 1, \ldots$

begin
  $k = 0, r = 0$ /* Discrete time and policy indices */
  $\eta = (n+m)(n+m+1)/2$
  Initialize $\mathbf{P}^{[0]}$ /* RH and positive definite */
  repeat/* Main timing loop */
    Find $\mathbf{e}_{k+1}$ using (4)
    Compute policy $\mathbf{u}_{k+1}^{[r]}$ using (10)
    if $[(k+1)$ modulo $\eta] == 0$ **then**
      $r \leftarrow r+1$/* Update policy */
      Solve for critic weights $\mathbf{w}$ using (11)
      Construct $\mathbf{P}^{[r]}$ from critic weight vector $\mathbf{w}$
      if $\|\mathbf{P}^{[r]} - \mathbf{P}^{[r+1]}\| < \varepsilon$ **then**
        Set $\mathbf{u}_{k+1}^{*} \leftarrow \mathbf{u}_{k+1}^{[r]}$
      else
        $k \leftarrow k+1$
    else
      $k \leftarrow k+1$
  until *Forever or tracking error is zero*

Figure: Model Free Reinforcment Learnging Algorithm

Model Free RL Algorithm

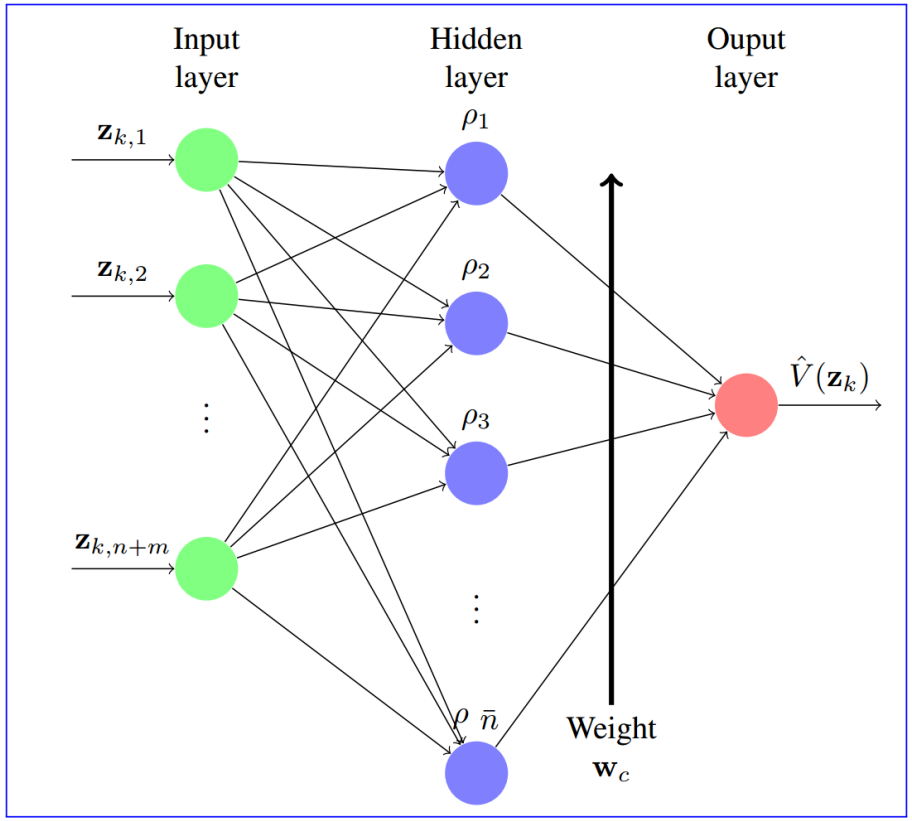


Figure: Critic Neural Netowrk Structure

Results

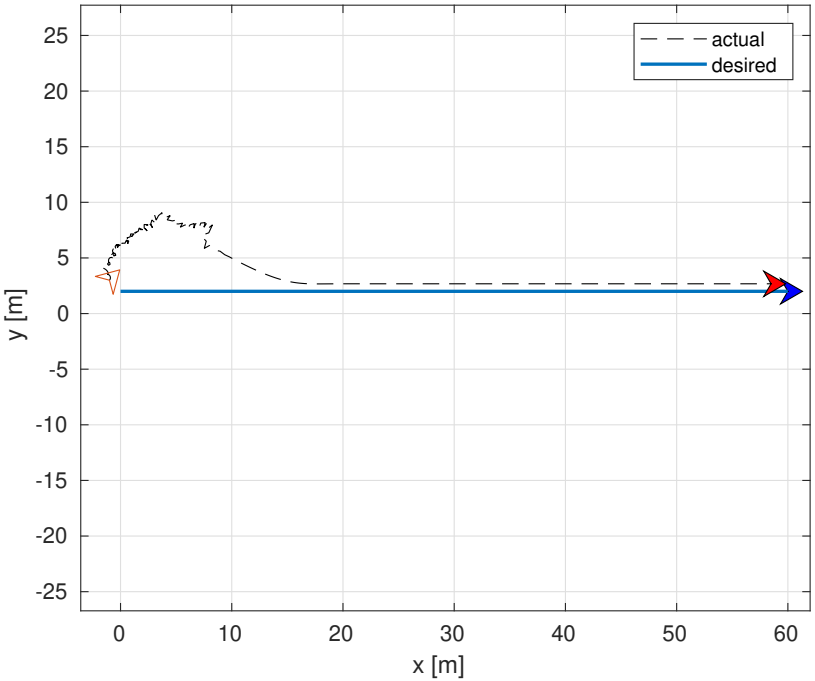


Figure: Trajectory

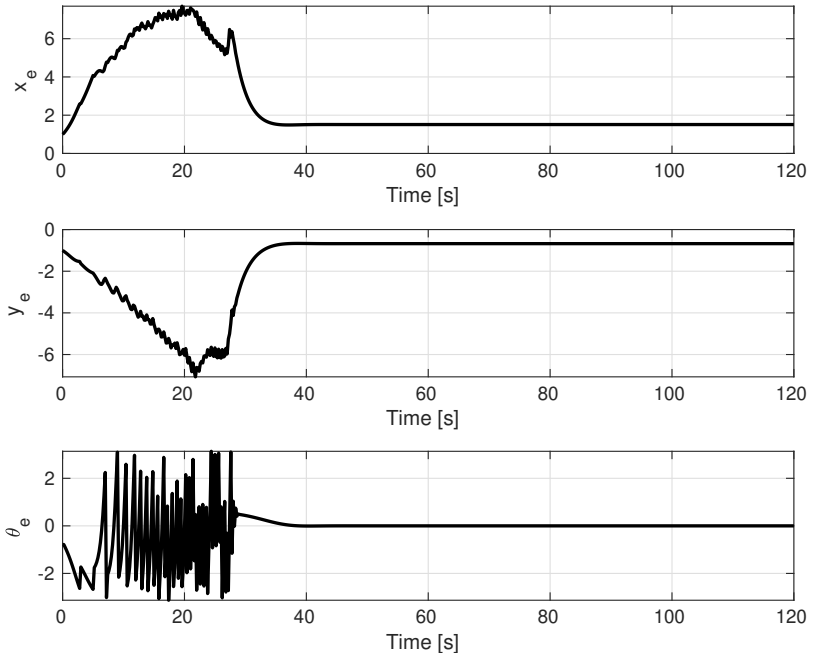


Figure: follower position error

Results

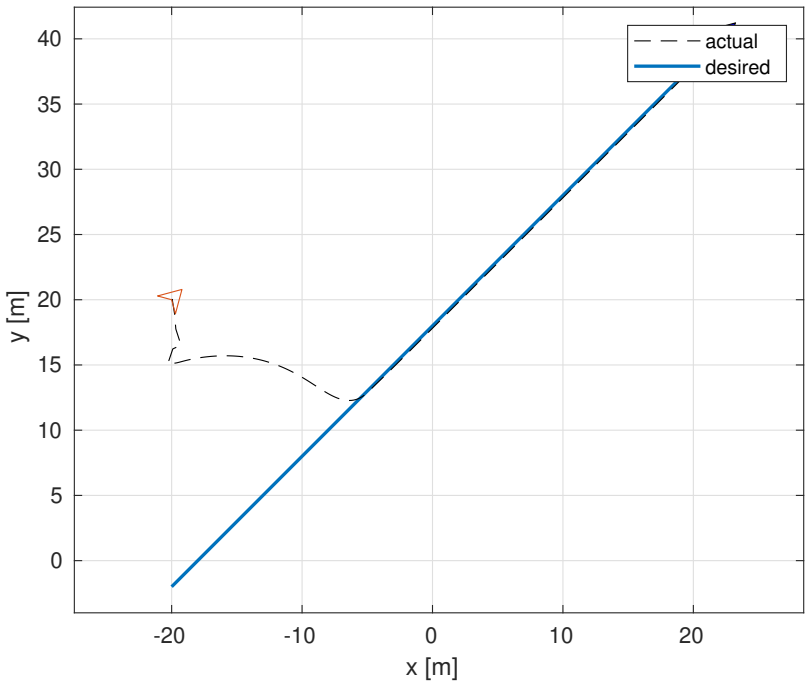


Figure: Trajectory

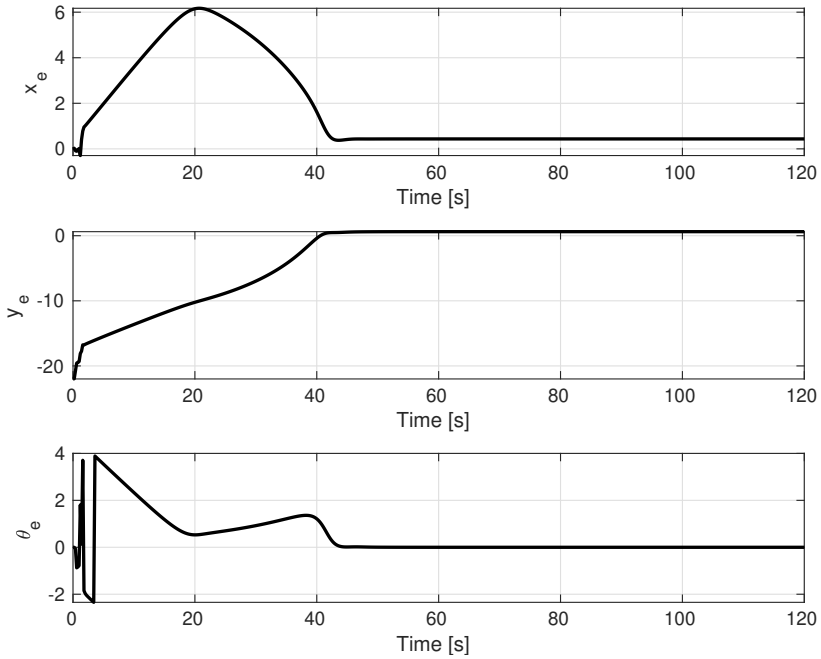


Figure: follower position error

Results

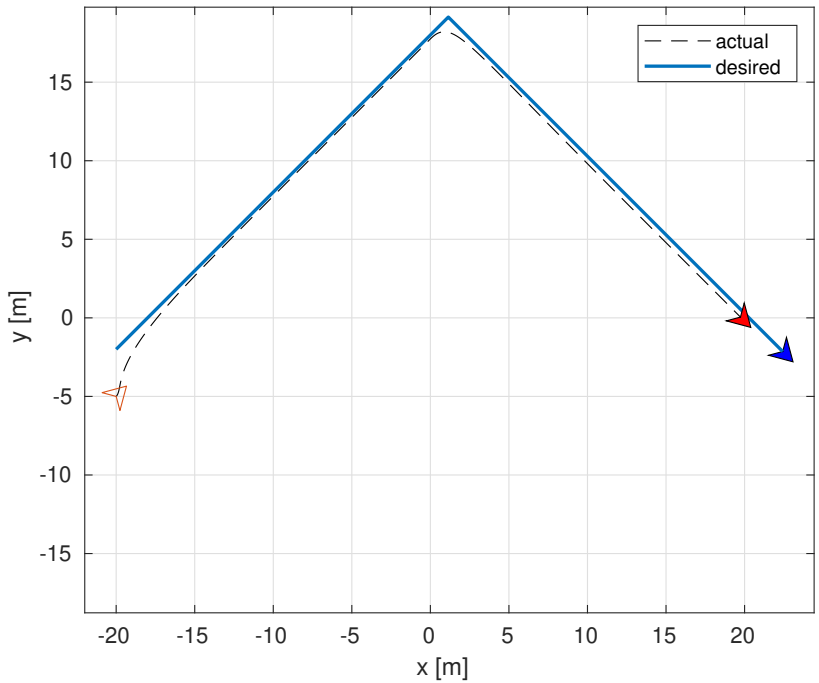


Figure: Trajectory

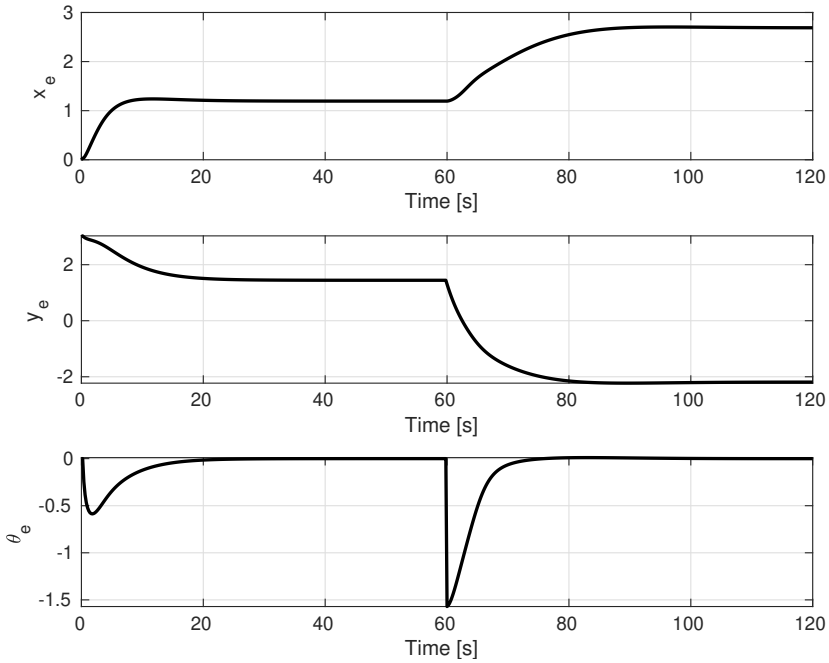


Figure: follower position error

Questions?