Cooperative Localization for Groups of Mobile Agents

(TurtleBot with Qualcomm Snapdragon ARM CPU)

UCI Research Project Under the Mentorship of

Professor Solmaz Kia

and

Professor Eli Bozorgzadeh

The presentation is made by UCI student **David Gogokhiya**

What Is the Purpose of This Research?

Develop a robotic testbed for a robot localization

Technique called **Cooperative Localization**

What Does **Cooperative Localization** Mean?

Finding your own position in the environment by

Sharing information between multiple objects

How Does **Cooperative Localization** Work?

Mobile agents take relative measurements between each other

Share this information between each other

Do computations to identify the position of every mobile agent

Get the updated positon

What We Used for Our Testbed?

We used multiple **TurtleBots** as the **Mobile Agents**

We replaced the netbook controlling unit of a TurtleBot with a **Qualcomm Snapdragon Microprocessor**

We used **Robot Operating System** (ROS) as our software Environment

Mobile Agent – TurtleBot

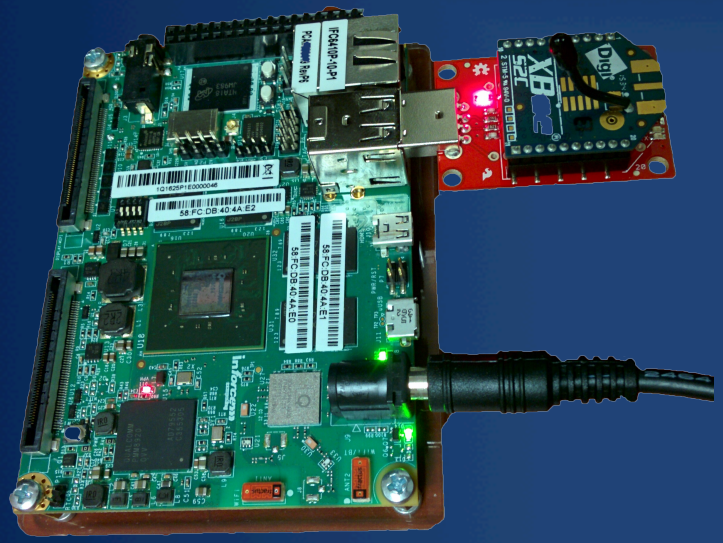
Low-cost robot especially made for Education and research purposes

---

Equipped with Kinect, a motion Sensing device

Create exciting applications using ROS and execute them on them a TurtleBot

Qualcomm Snapdragon ARM CPU



Powerful microprocessor

Located on a single board Computer

High performance

Robot Operating System (ROS)

Collection of frameworks to control Robots

Easy to learn

No need to reinvent the wheel – Don't Code what was already coded for you

Open source

Combining all these components we are able

To execute the **Cooperative Localization**

Algorithm and prove its efficiency

… But why do we need it?

Why don’t we use GPS?

GPS?

It is not always possible to receive Persistent GPS signal

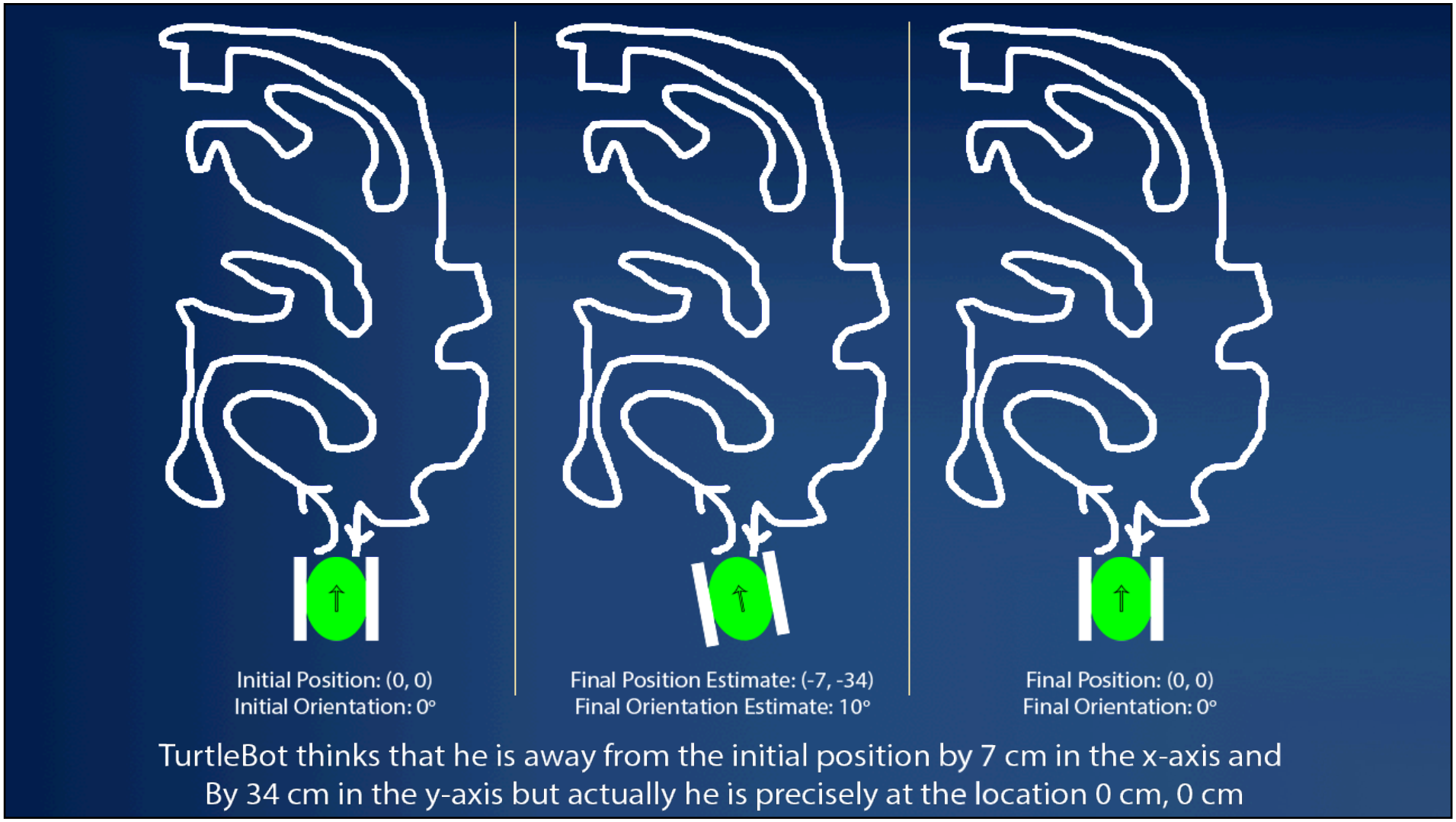
GPS is not very accurate

GPS doesn’t work properly inside the Buildings

Furthermore, based on an experiment that we performed,

After 3.5 minutes of navigating the TurtleBot in a chaotic path we

Observed a **30 cm** error in a robot's location estimate

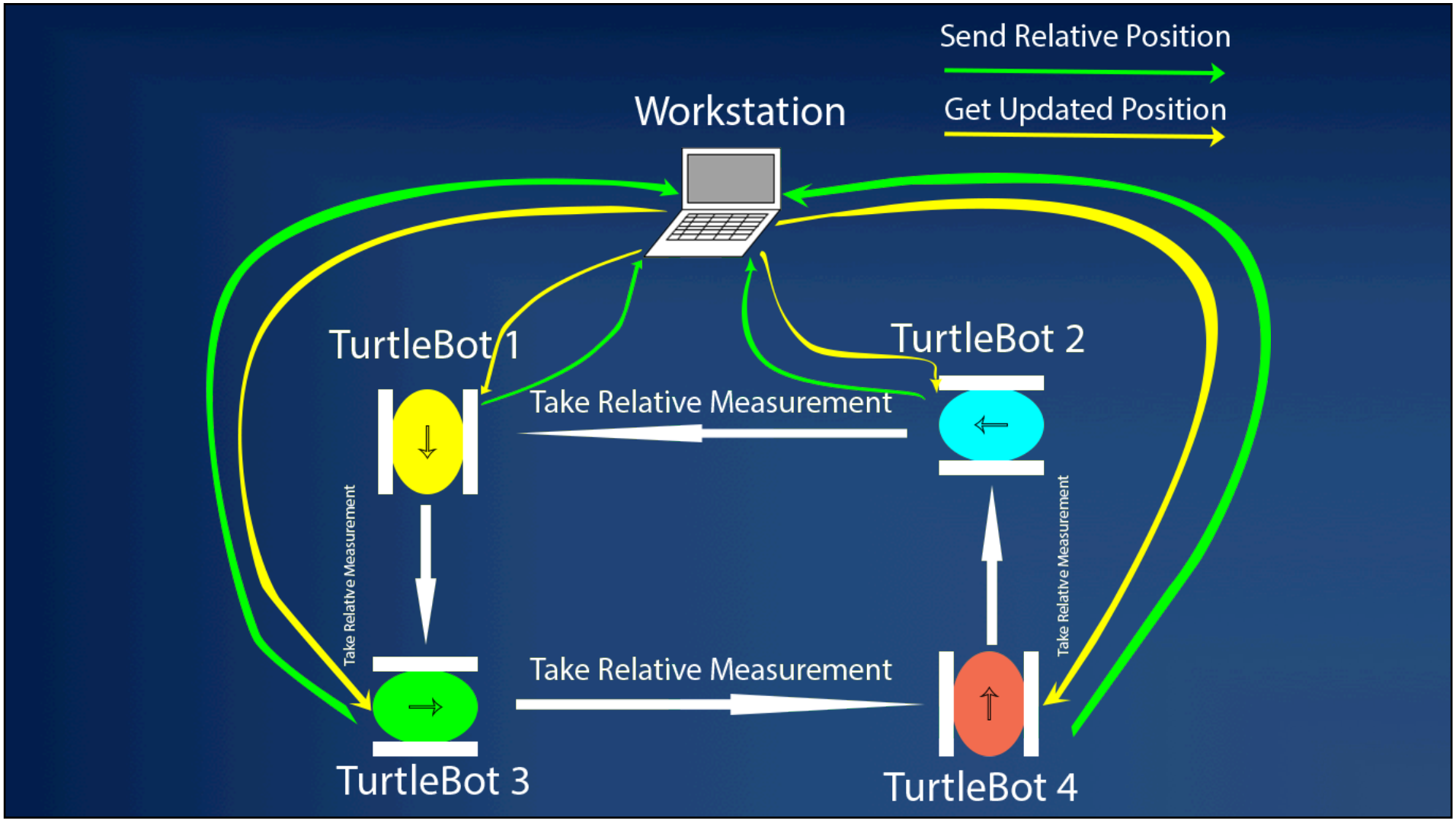

Initial Position: (0, 0)
Initial Orientation: 0°
Final Position Estimate: (-7, -34)
Final Orientation Estimate: 10°
Final Position: (0, 0)
Final Orientation: 0°
TurtleBot thinks that he is away from the initial position by 7 cm in the x-axis and
By 34 cm in the y-axis but actually he is precisely at the location 0 cm, 0 cm

Therefore, we have to come up with a different technique

Of how to localize the robot

Cooperative Localization is a perfect solution

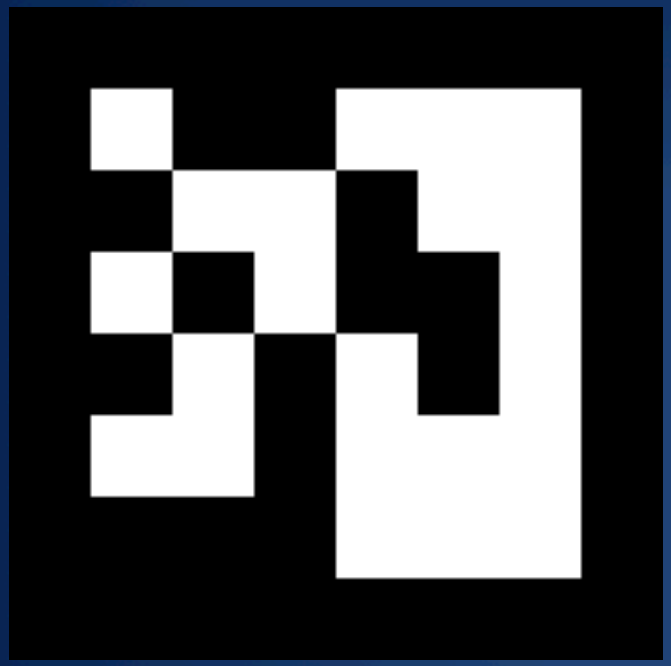
How We Developed Our Testbed?


Send Relative Position
Get Updated Position
Workstation
TurtleBot 1
TurtleBot 2
Take Relative Measurement
Take Relative Measurement
Take Relative Measurement
Take Relative Measurement
TurtleBot 3
TurtleBot 4

How Do We Take the Relative Measurements?

We used **Kinect** as our motion sensing device to

Detect other TurtleBots



… In order to distinguish

TurtleBot from any other

Obstacle we used **Ar Tags**

We have created an **Ar Tag Cube**

… And we put this

Cube on each TurtleBot

In order to prove that

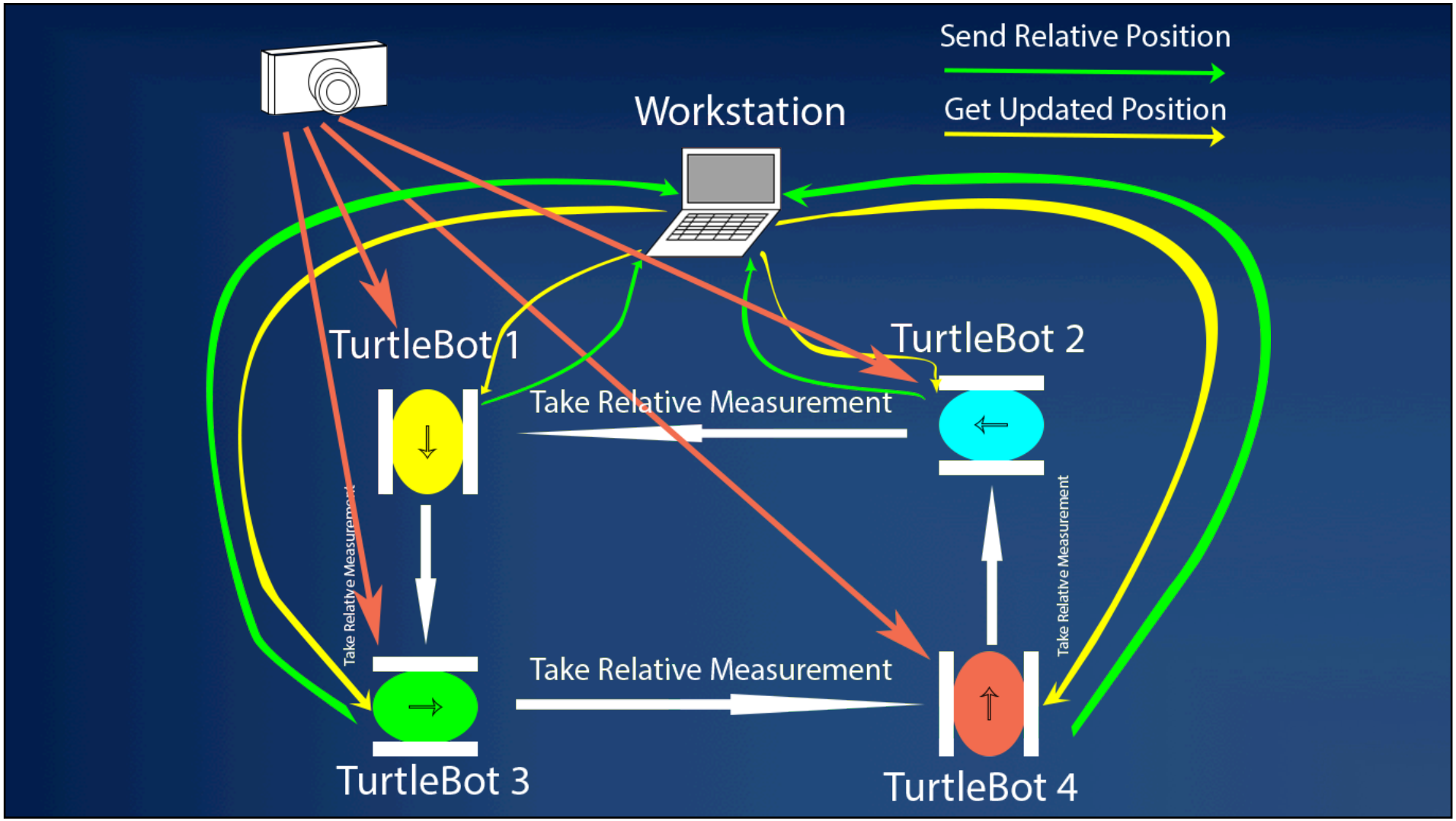
Our algorithm works we

Used an additional

Camera as a reference

Camera is mounted to the Ceiling

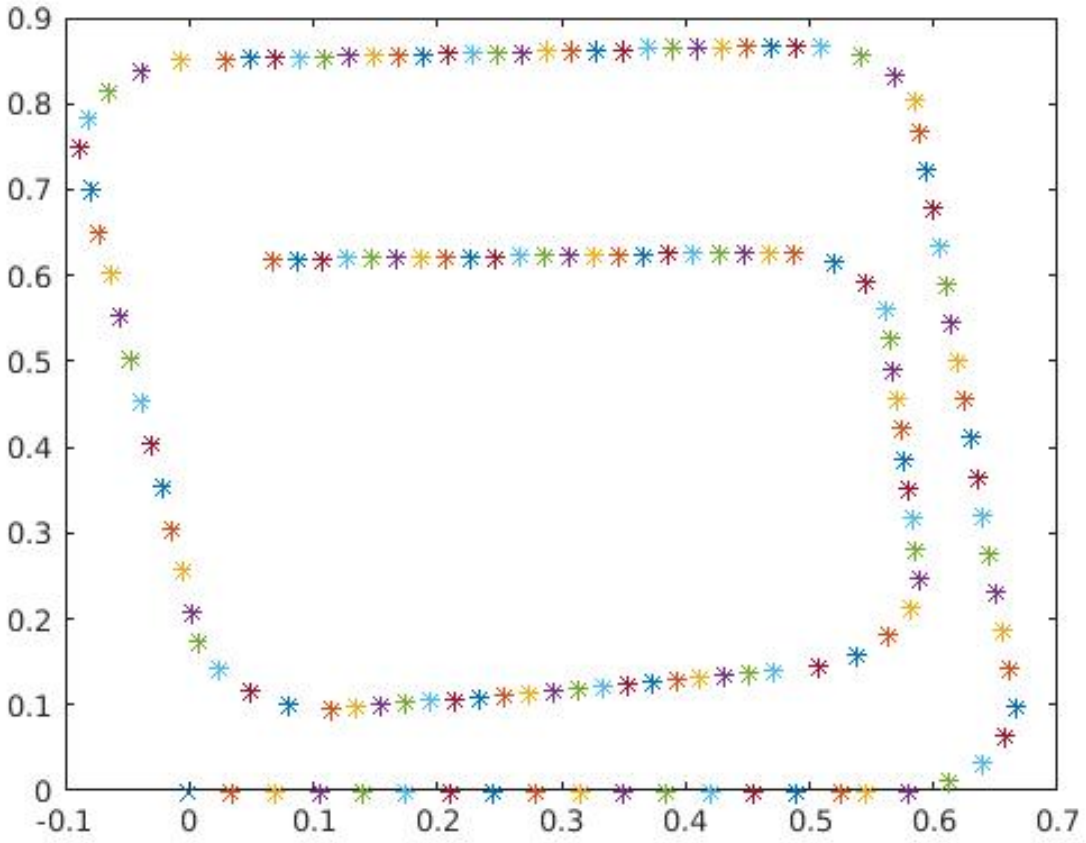
It detects the TurtleBots Based on the unique Ar Tag Cube located on every TurtleBot

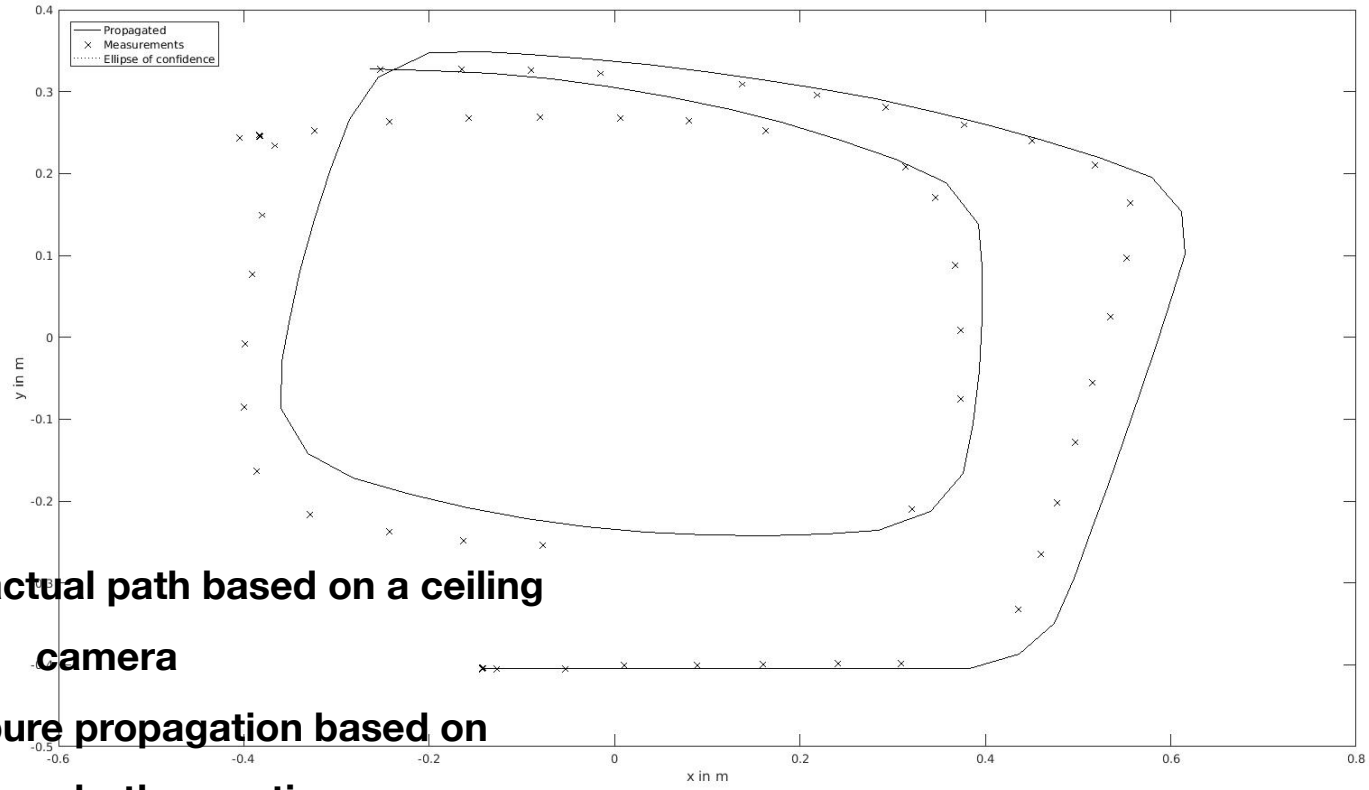

Send Relative Position
Get Updated Position
Workstation
TurtleBot 1
TurtleBot 2
Take Relative Measurement
Take Relative Measurement
Take Relative Measurement
Take Relative Measurement
TurtleBot 3
TurtleBot 4

In order to see the deviation in a path we

Also had to create a script to move the

TurtleBots in the predefined path


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7



x <– actual path based on a ceiling camera

– <– pure propagation based on robot's equations

Next Steps

Until the end of this week we plan to perform a test run with four Robots to see how efficiently our algorithm works

After that we plan to make a test run when one of the TurtleBots Misses multiple messages with the updated position

Future Work

Implement another more efficient algorithm

Make our system fully distributed – remove the workstation from the system to make it more reliable


And a Small Demo
in the End