# TurtleBot in the Cloud

Divya James Athoopallil
Sravanth Yajamanam

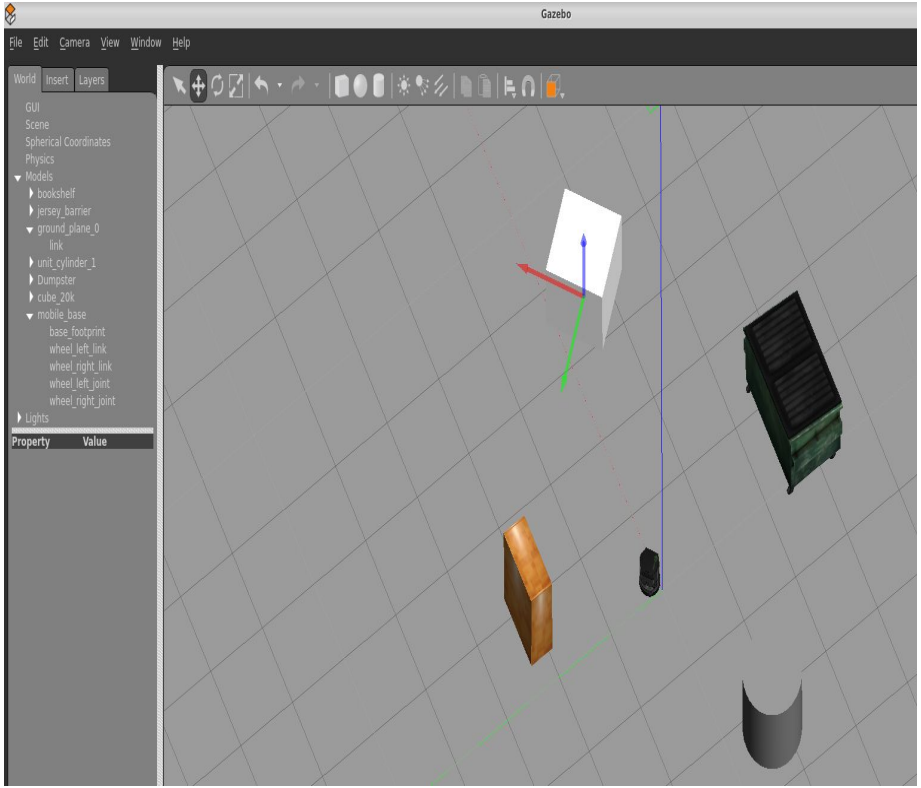
# PRESENTATION OUTLINE

- Project Goals
- Software components
- Architectural diagram
- Testing and Debugging
- Capabilities and Limitations
- Learning and outcomes
- Demo

# ROBOTIC SYSTEM & CLOUD INFRASTRUCTURE

- Tasks in robotics have ever changing resource requirements.

- Cloud computing allows for flexible allocation of compute and memory,  facilitate real-time deployment.



# EXPERIMENT

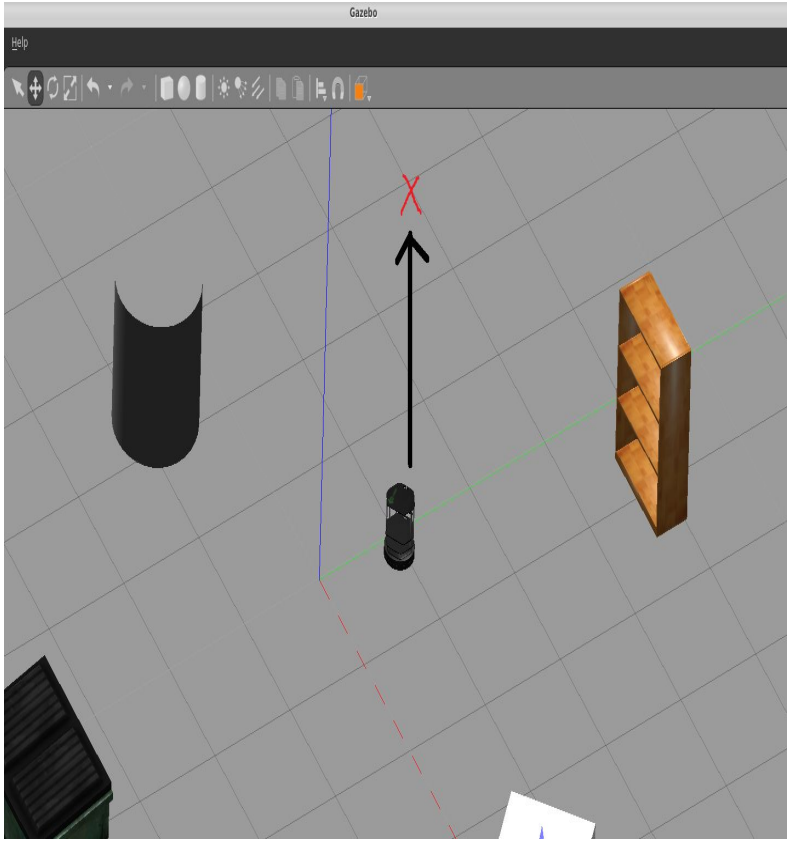
Move  robot to move from point A to B and pause if obstacle encountered.

We have used a robot called TurtleBot in the gazebo simulator.

## Task

Estimate Turtle current location, plan a path and move to the given destination.

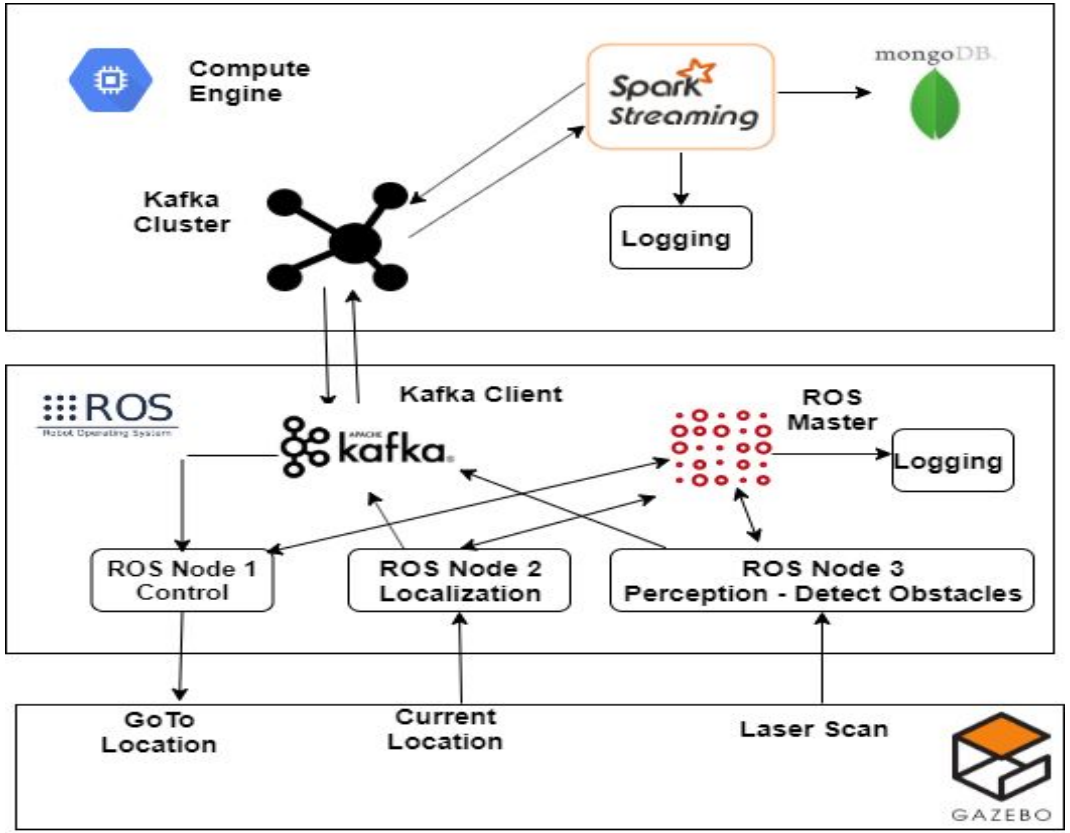
If the robot encounters obstacles along the way, the robot pauses until the obstacle is moved.



# COMPONENTS

- Robot Operating System(ROS), A peer-to peer distributed system.
- Gazebo, 3D robotic simulator and User Interaction
- Zookeeper, cluster management
- Kafka, messaging
- Spark Streaming, compute/process engine
- MongoDB, database
- Google Cloud Platform

# ARCHITECTURE




Architecture Diagram

# TESTING & DEBUGGING

## Unit and Integration Testing

- Kafka producer/consumer testing
- Spark-Streaming Kafka integration and testing
- Spark MongoDB integration and testing
- ROS Kafka Bridge publisher subscriber testing

## Debugging

- ROS logger (used default log for “info” , “debug” & “WARN”)
- Spark Streaming logger (using log4j properties conf file)

# CAPABILITIES

- We can scale this project to multiple robots for the similar tasks i.e motion planning.
- Spark Streaming can scale computation across different robots.
- Handling varied latency eg. sensor with different latency when known in advance.
- Storing data in MongoDB allows performance analysis of different path planning algorithms.

# LIMITATIONS

- TurtleBot has several functionalities that we haven't implemented, but our current framework can be expanded to accommodate them.
- In order to accommodate more diverse requirements it would be better suited if we had implemented consumer groups in kafka.
- We can implement better visualization with the data collected in MongoDB making analysis easy

# LEARNINGS

- Implementing the most basic task was cumbersome due to the lack of resources.
- We referenced multiple journal articles to evaluate the pros and cons of selecting each component.
- Understanding the implementation of each component individually and designing an optimal architecture to integrate ROS and Cloud Infrastructure
- Manipulating distributed architecture of ROS
- Working with gazebo
- Kafka/Spark Stream Integration

# WORKING DEMO:

# Thank You!