Fetch

ENPM 808K Human Robot Interaction

Final Project.

Fall 2017

Yash Manian
Kavit Patel
Chinmay Rao

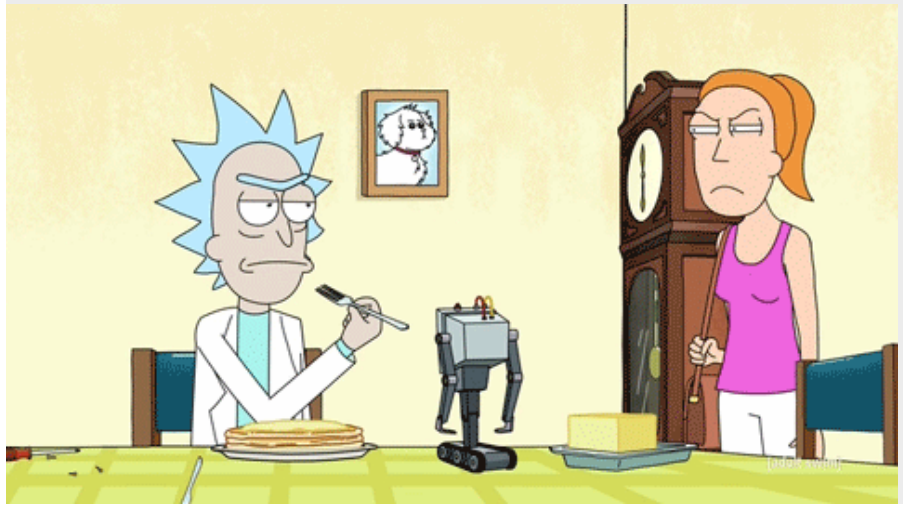
University of Maryland
Robotics

Course Instructor: Josef Schaff

Operation guidelines

- The user should be able to summon and command the robot using only speech input.
- The robot should identify the object required from either speech or input from the display.
- The robot should accurately be able to navigate to and retrieve specified object. If object is not found, the robot is to return to the user.
- The robot should learn over time, where objects are most likely to be found.
- Should be flexible enough to learn new objects and locations.
- Should be flexible in terms of the hardware being used.

Target Demographic



- The disabled and elderly populations.
- People who cannot move from a spot for long hours (Students, programmers and gamers).
- Lazy scrubs.
- Tech savvy salty grandpas.
- Pretty much everyone.

Fetch Overview

3D RGB Depth Sensor
with Microphone

Speakers

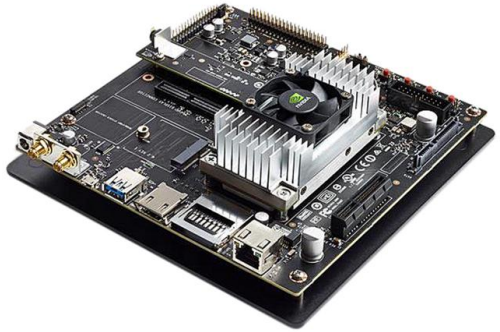
Display

Gripper

5 DOF Arm

Drive Controller

Ultra-Sonic Sensors

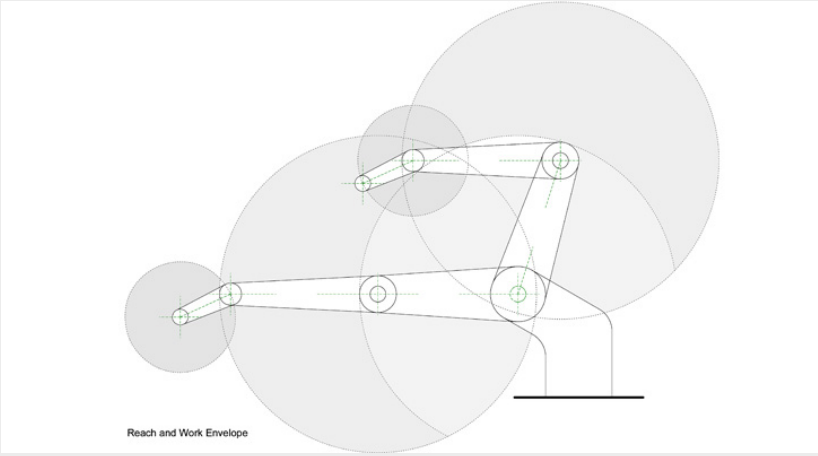


Remote Controller

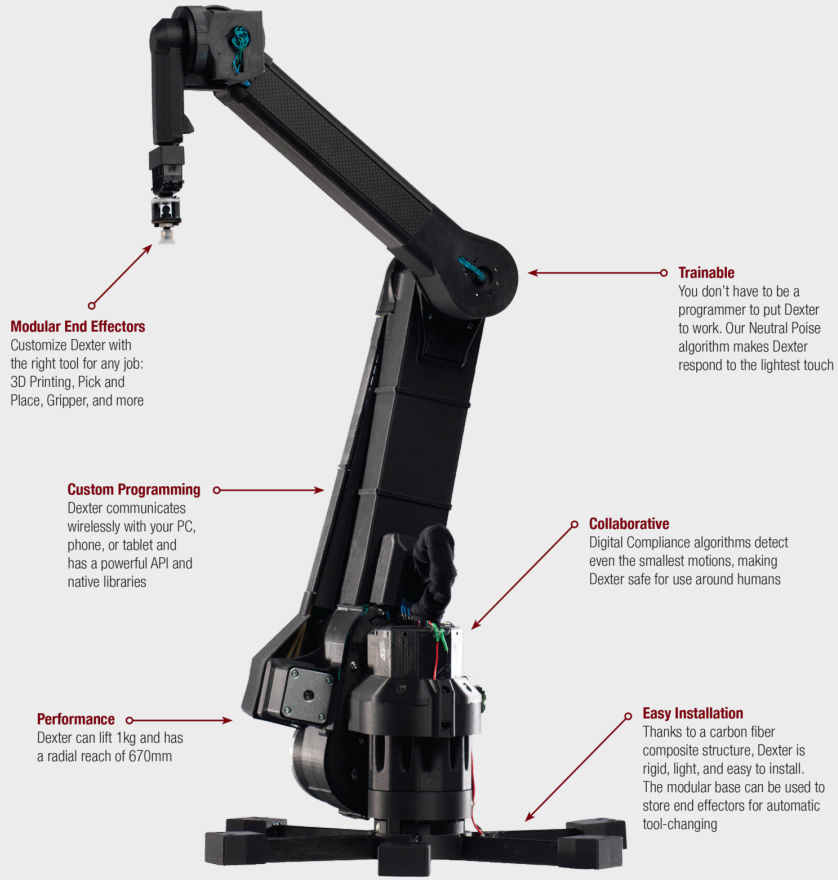
Fetch Specifications

- **Fetch Robot**
  - Weight: Approx. 14kg
  - Height: 1000mm
  - Base: 400mm diameter
  - Max Speed: 0.5m/s
  - Environment: Indoors
- **5 DOF Arm**
  - Payload: 1kg
  - Arm Length to Gripper: 670mm
- **Sensors**
  - Ultrasonic
  - 3D RBG Depth
  - Odometry Encoders
  - Microphone
- **Capabilities**
  - Speech Recognition
  - Localization
  - Object Detection
  - Trajectory Planner
  - Visual Servoing and Grasping
- **Applications used**
  - ROS Navigation
  - MoveIt!
  - Calibration

Robotic Arm- Dexter



- Weight: 6.5kg
- Power: 45W
- Arm Length: 670mm
- Pay Load: 1kg
- Connectivity: USB, Wi-fi, 10bit GPIO
- DOF: 5+
- Power consumption: 20 W at no load, 60 W at Max load.
- Precision: 50 micron



Robot Drive

- Dimension: 363*363*140mm
- Load Capacity: 15kg
- Net weight: 5.4kg
- Speed: 0.6m/s
- Power:17w
- 6 Ultrasonic Range Finders
- 3 IR sensors
- Arduino 328 Controller

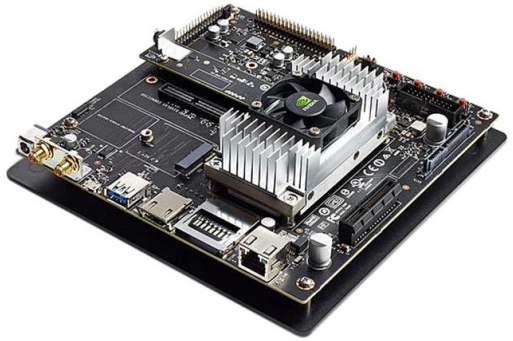
System Architecture

Camera , Depth & Microphone

Display

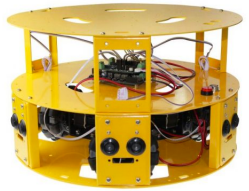
Speaker

Arm Controller



Remote Controller

Onboard controller



Drive Controller

Wi-Fi

ROS

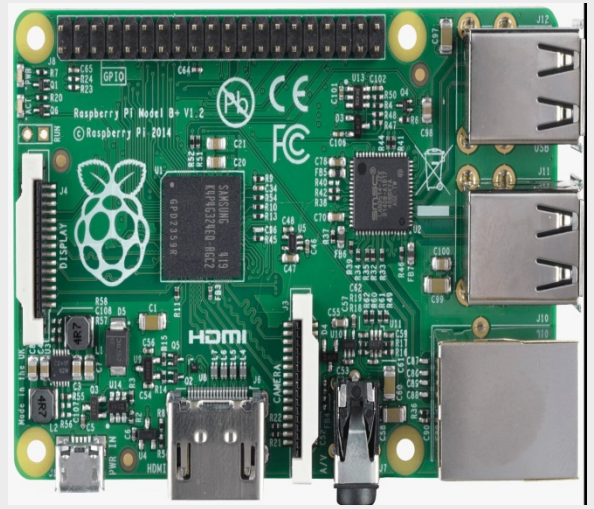
Drive Sensors

Jetson TX2

- **Specifications**
  - GPU: NVIDIA Pascal 256 CUDA core
  - CPU: HMP dual denver + Quad Arm
  - Memory: 8Gb
- **Applications-High Level Control**
  - Object Detection and Classification
  - Trajectory Planning
  - Localization

RaspberryPi 3

- **Specifications**
  - CPU: 1.2 GHz Quad core ARM cortex A53
  - GPU: Broadcom VideoCore IV @ 400 MHz
  - Memory: 1 Gb
- **Applications - Low level control**
  - Robotic Arm control
  - Drive Control
  - Elementary sensor data processing



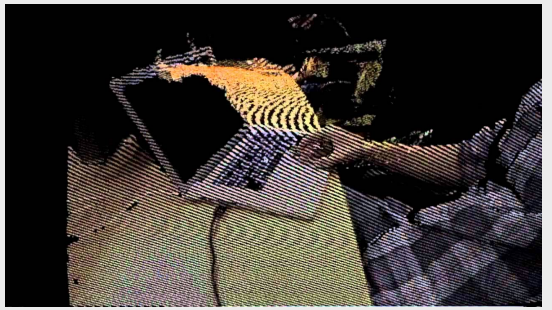
Intel RealSense SR300

- **Specifications**
  - Camera: RGB
  - Video: 1080p @30fps
    720 p @60 fps
  - Depth range: 20 cm – 150 cm
  - Microphone: Dual array microphone
- **Outputs**
  - 3D images of its surroundings
  - Point cloud data
  - Acoustic data

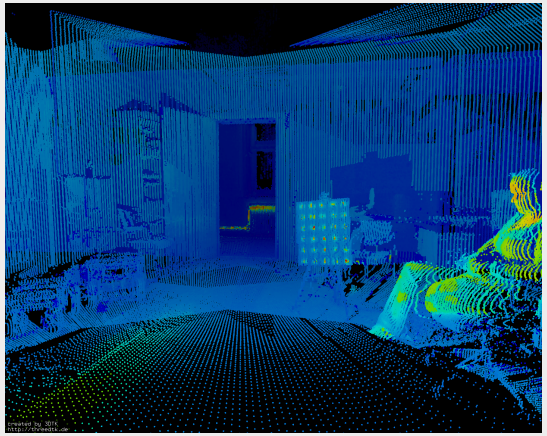
Other Components

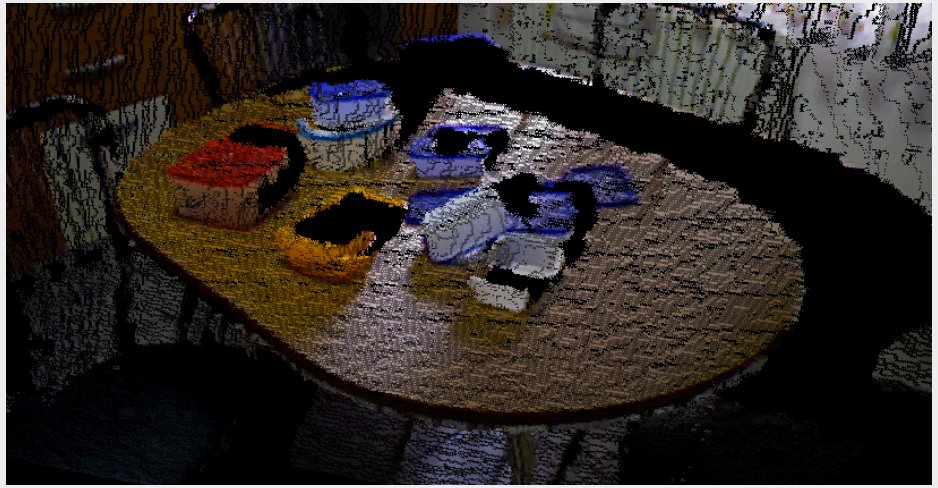
- **Batteries**
  - **Electronics battery**
    - Battery type: LiPo
    - Voltage: 11.1V
    - Capacity: 2000mAh
    - Peak Discharge: 20C
  - **Robot Arm battery**
    - Battery type: LiPo
    - Voltage: 22.2V
    - Capacity: 10000mAh
    - Peak Discharge: 20C
- **Display**
  - Size: 7 inch
  - Multitouch: Capacitive
  - Connectivity: USB 3.0
- **Speaker**
  - Model: Adafruit speaker mini
  - Wattage: 3 W

Point Cloud Data

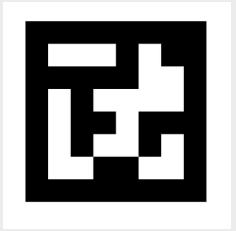


- Point cloud is a set of large number of data points in the view of the sensor measured by 3D scanners.
- This data can be used for localization and geometry identification.

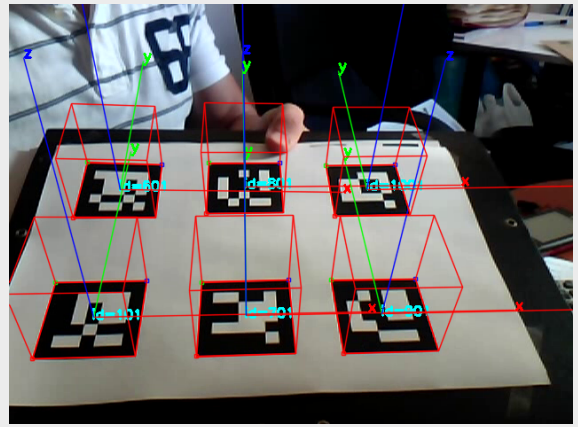


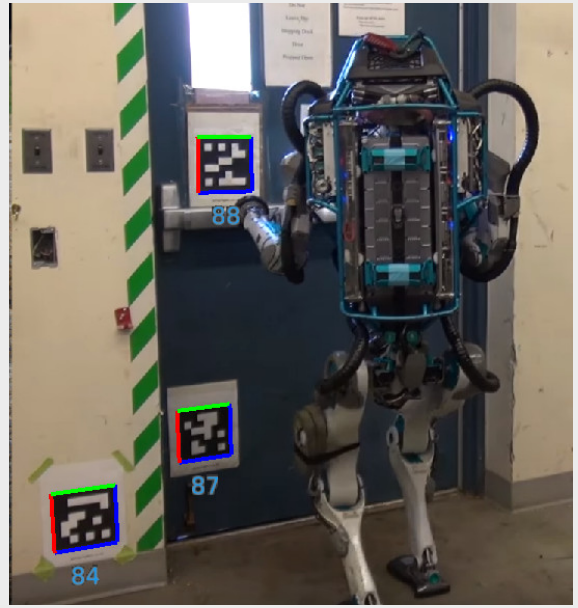


AprilTags Detection for Localization



- AprilTags is a visual fiducial system.
- AprilTags detection software computes the precise 3D position, orientation, and identity of the tags relative to the camera.
- Outside every room a pair of them to localize better and remove the error due to drift over time.



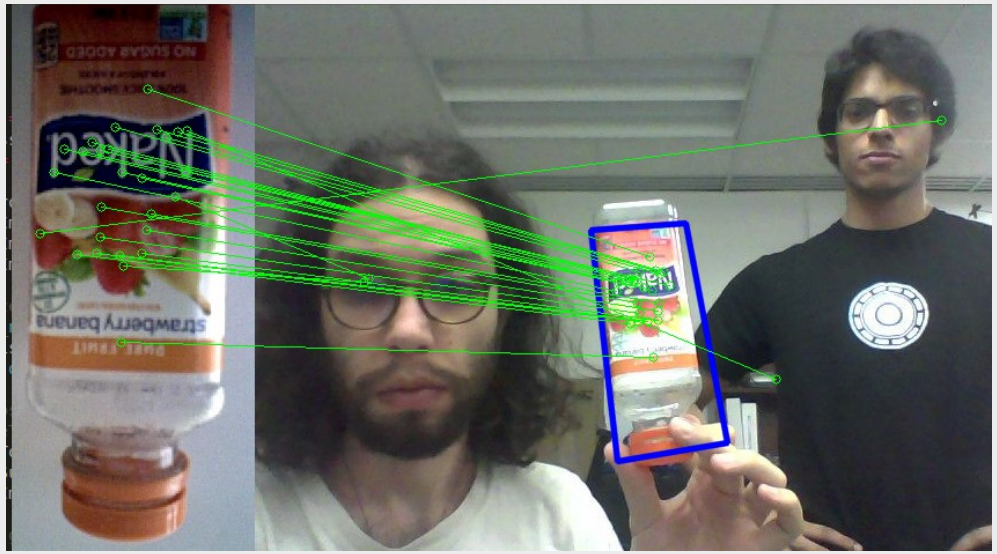


Object Detection and Classification

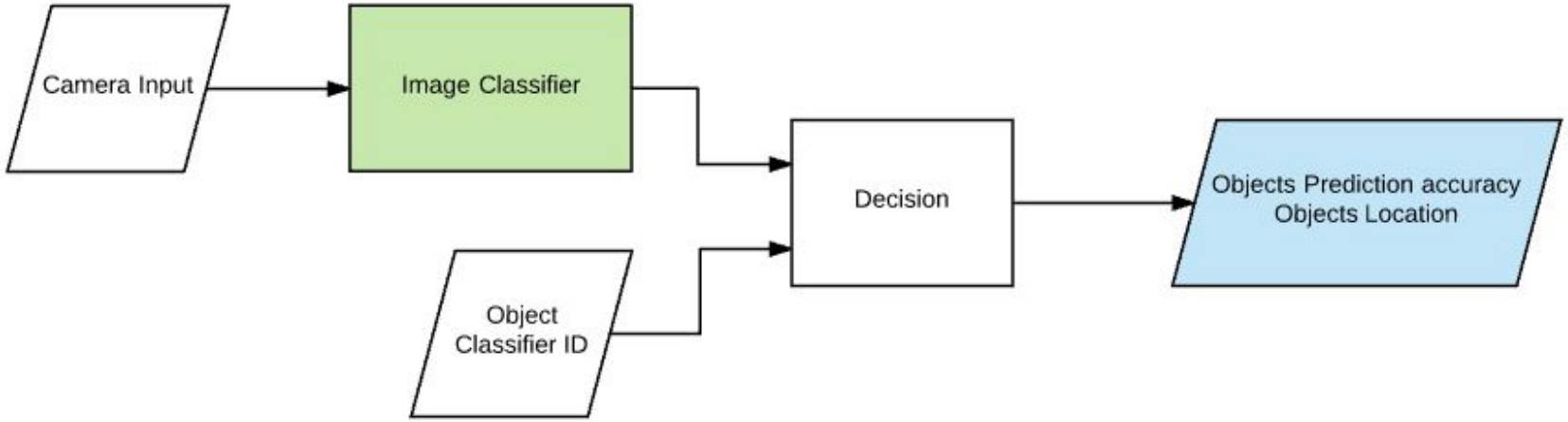
- Images are broadcasted to the remote controller via ROS interface.
- The pre trained classifier system uses feature selection algorithm to identify objects in the image.
- These objects are classified using ‘The household_objects SQL database’ with a prediction accuracy and multiple grasp orientations.
- This database consists of images from online outlets like Ikea and Target, and common objects such as bowls and plates.

Training based on SIFT features

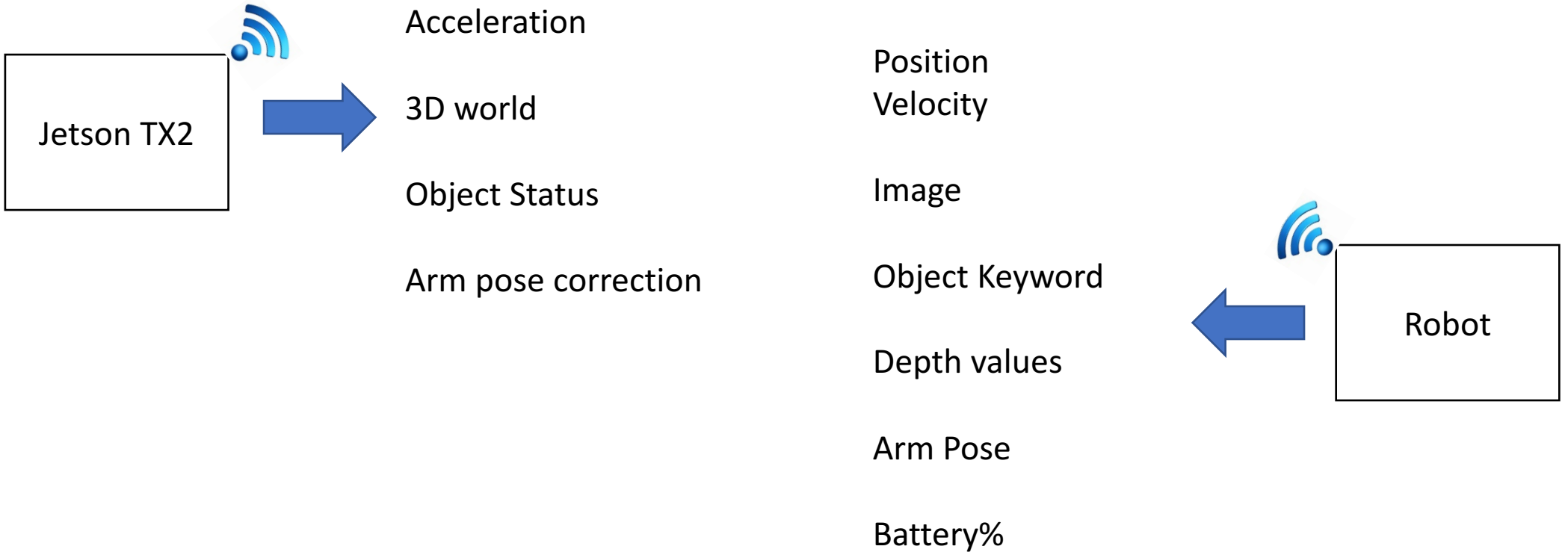
- SIFT features for every object in the training set are used as inputs to CNN.
- This reduces the number of features to be trained on.
- This in turn reduces classification time during operation.



Object Detection and Classification



Interaction between Remote Controller and Onboard Controller



Assumptions

- The environment has adequate illumination for object detection and classification.
- The probability of finding the object in the map is 1.
- The object is within the acceptable payload limits of the robot.

Autonomy and Learning

- Requires the ability to identify keywords from speech
- Should be able to determine most likely location of object being sought
- Should have the ability to navigate to said location
- Should be able to identify and procure object of interest. If it can’t find the object, look in the next most likely location
- Should be able to return the object to the user.

State Machine

1

Speech

- Process speech input

2

Localize

- Determine likely location of object

3

Planning

- Plan trajectories to all locations in order of likelihood

4

Identification and classification

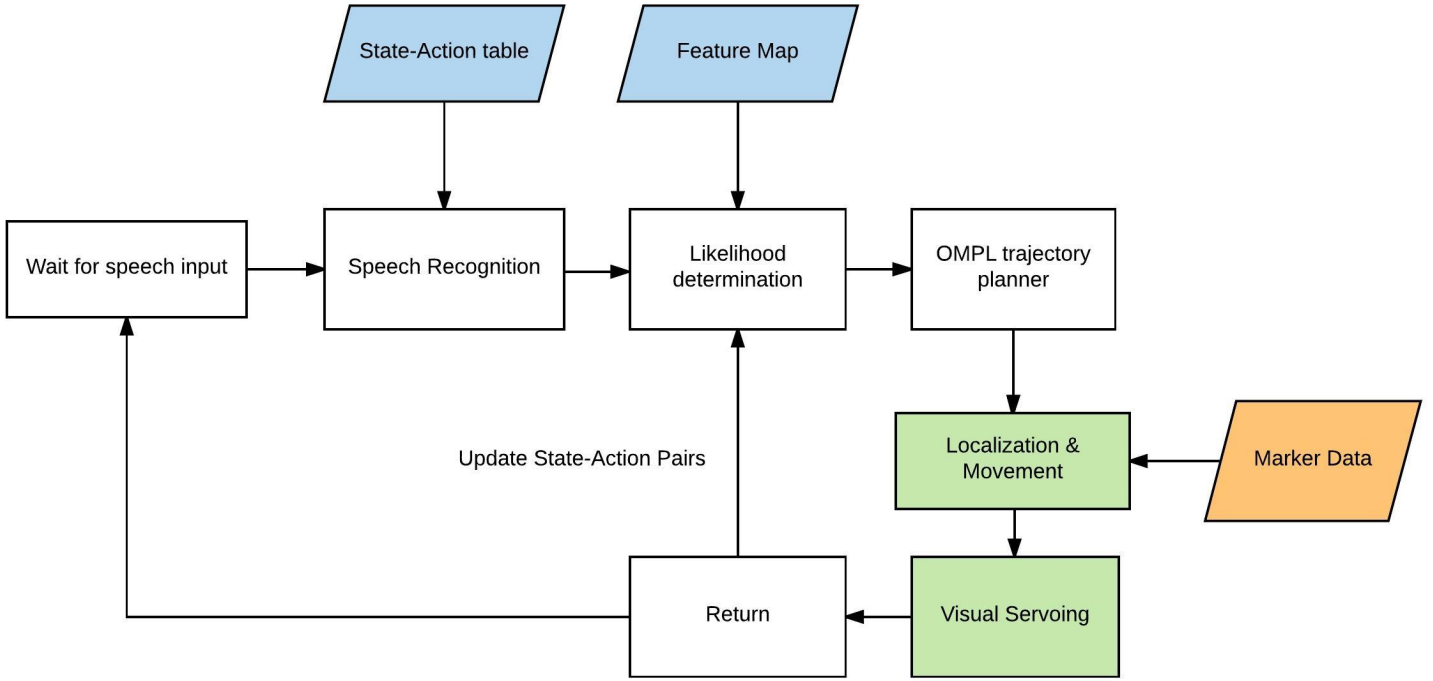
- Once at location, identify and classify object

5

Procure

- Grasp object and return to user

State flow

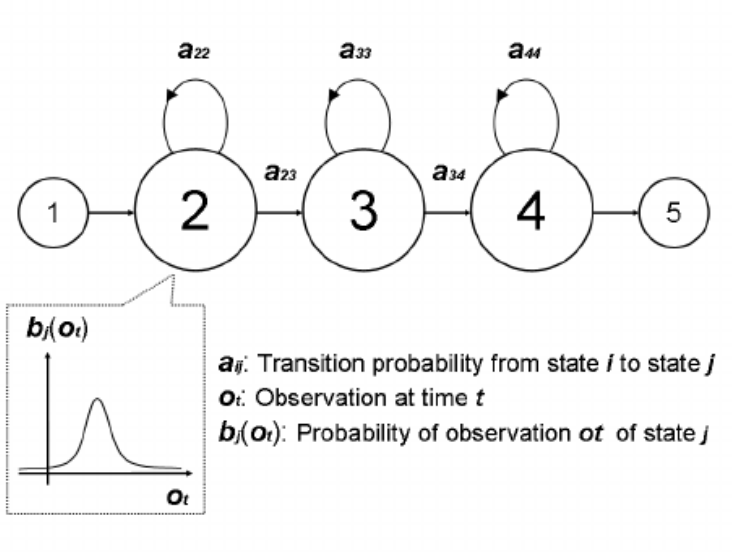

State-Action table
Feature Map
Wait for speech input
Speech Recognition
Likelihood determination
OMPL trajectory planner
Update State-Action Pairs
Localization & Movement
Marker Data
Return
Visual Servoing

Speech Processing

CMU Sphinx

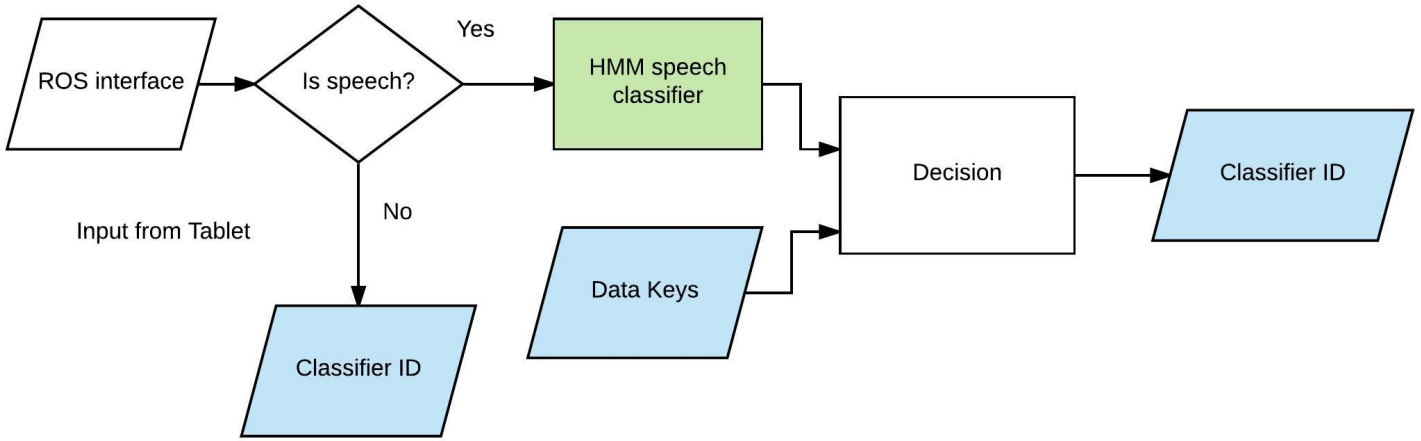
- Voice data moved over ROS interface with the google_glass_driver package.
- Using the CMU Sphinx library to recognize keywords in speech.
- Generate Classifier ID for object based on keyword recognized.
- The information extracted is an attribute, and an object, hence commands such as a “red bowl” can be given to the system.

Hidden Markov Models



- Speech pattern recognition using hidden Markov models
- Maximizes probability of a word sequence given acoustic data.
- Computes probability at phone and word level then concatenates to form word sequence.

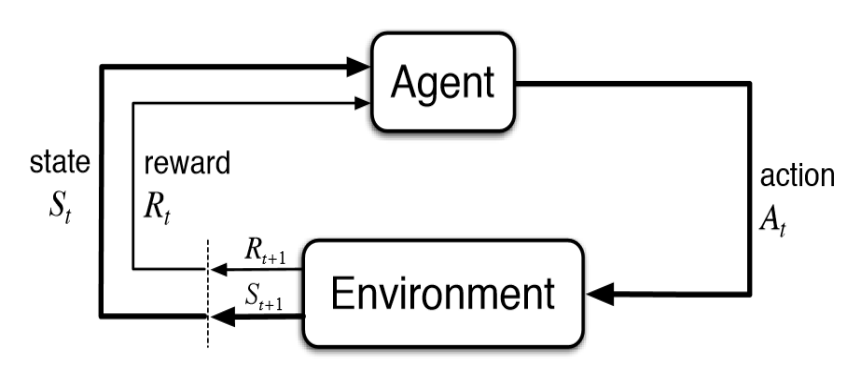
Speech Processing



Learning Locations of Objects

- Use Reinforcement learning to continuously learn the locations of objects in the residence.
- The State-Action table is updated every time an episode, i.e. a complete cycle of fetch has completed, where the likelihoods are assigned to each location for every object.
- The SA table starts off with pre-defined probabilities.
- Every time the system is trained on an new feature or object, an entry is added to the SA table.

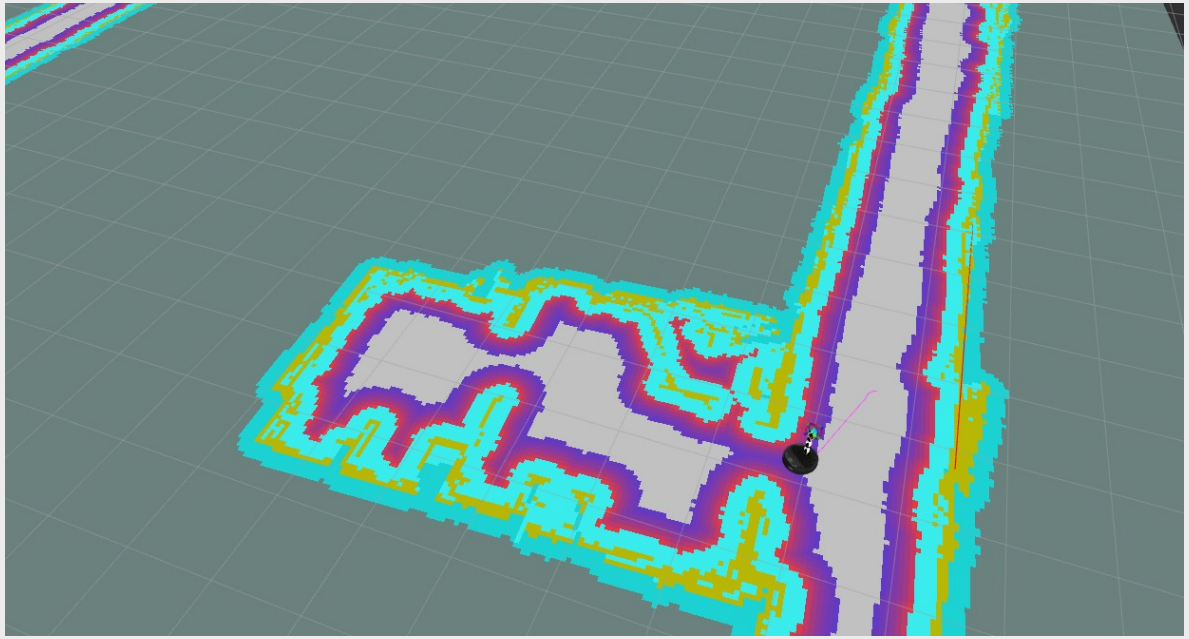
Reinforcement Learning



- Learns through every request. All rooms have continuously variable probabilities.
- Has the tendency to follow the highest likelihood, But there is also an ε-factor, which makes the robot explore from time to time.
- Rewards assigned to locations where desired object is found.
- The locations are assigned priorities as a function of distance and likelihood of finding the object there.

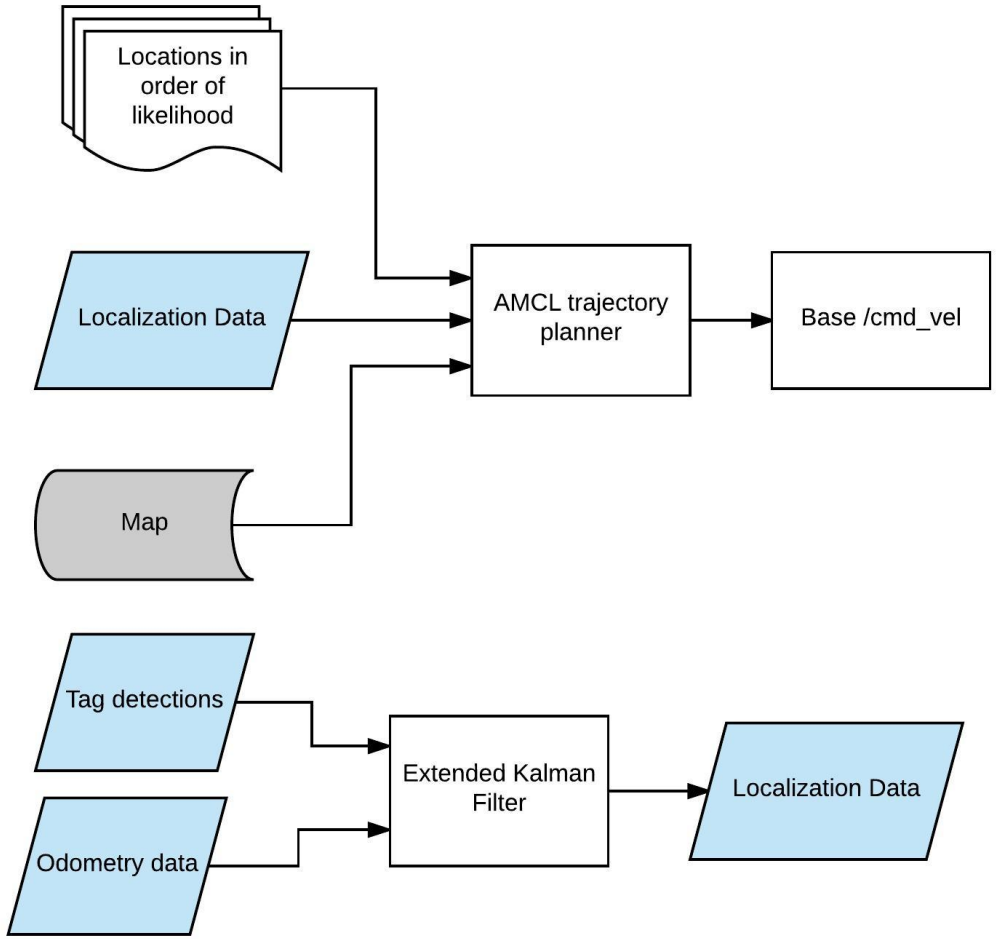
Planning and Navigation

- Uses the ROS moveit library to plan a path to location on a 2D map.
- Localization is done with the help of encoders and onboard depth sensor.



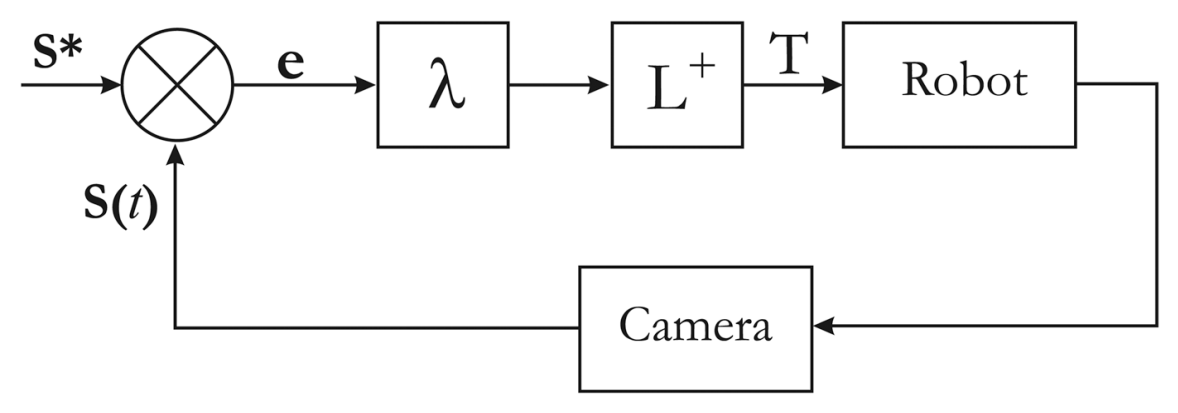
- AprilTags are used to improve accuracy to get through doors and identify rooms.

Trajectory planning and Localization

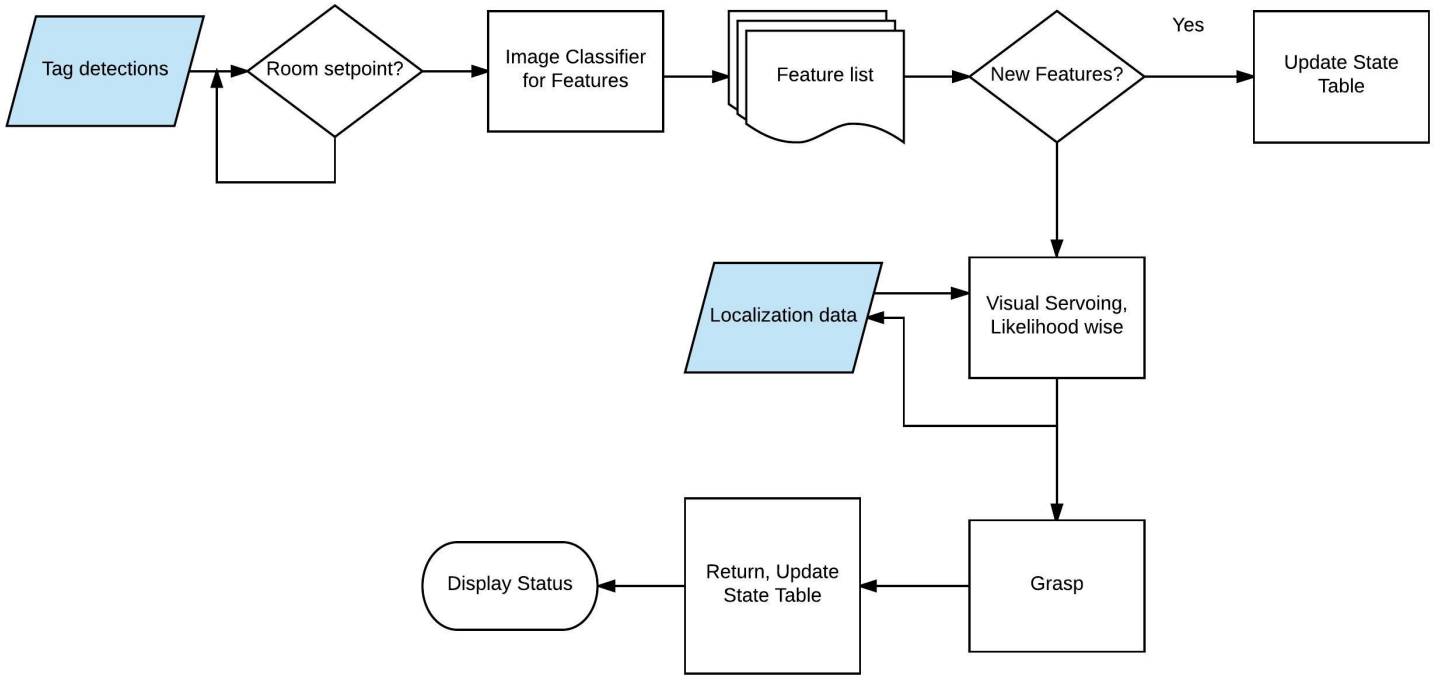


Grasping

- Once the robot has entered the room, search mode is entered.
- The features in the room are identified and localized.
- Uses visual servoing to grasp the object and return.



In-room State



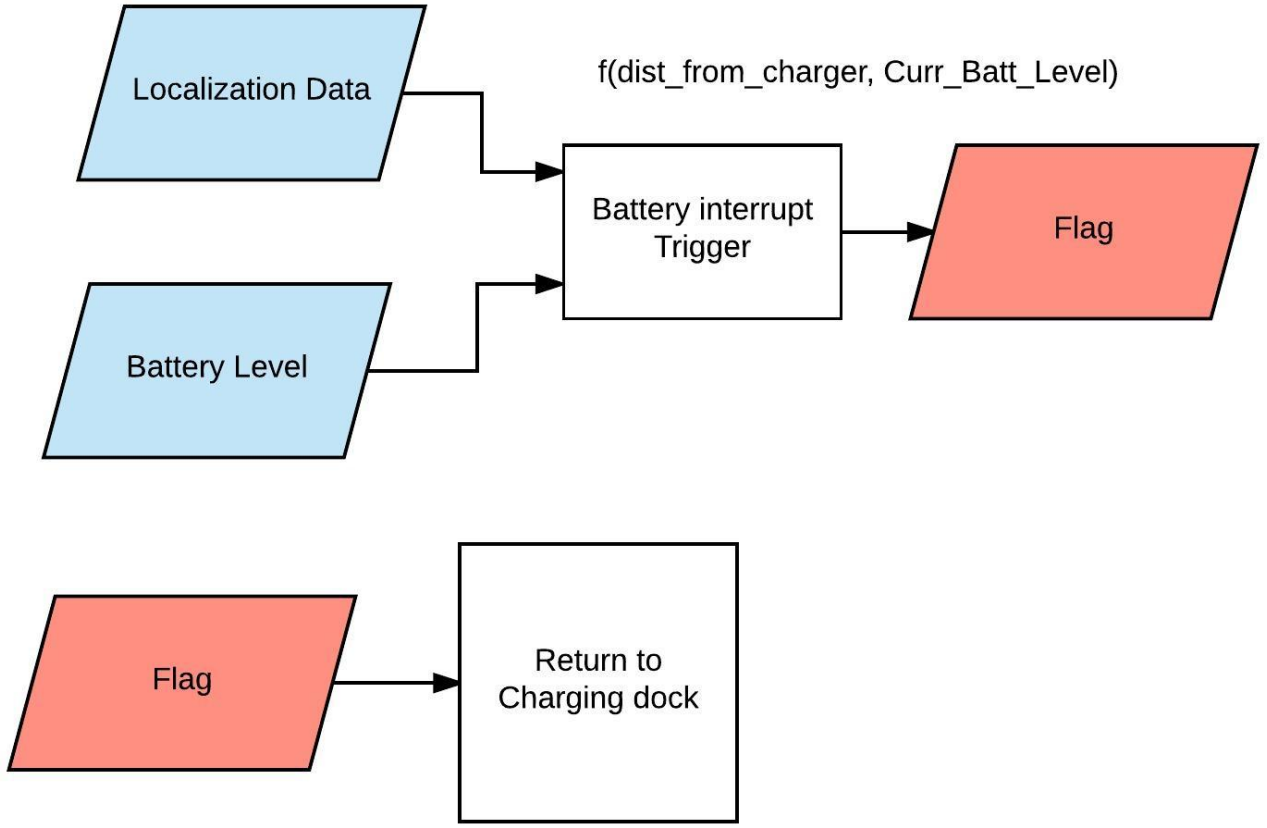
Location Training

- The system uses lifelong reinforcement learning to train itself over time. This means, that the robot’s performance keeps getting better over time.
- As a part of the installation service, the house will need to be mapped and tags will have to be installed outside every room.
- This also covers the assignment of initial probability values.

Object training

- The objects are trained on based on the libraries provided. However, there is a special training mode for uniquely shaped objects, where the point cloud is used to construct a model of said object.
- Only a limited number of special objects can be trained on.
- While classification, the object classifier ID is used to decide between the neural network or the point cloud models.
- A user mode will be available to increase the probability of objects being found in certain areas. This will be used to update the State table.

Battery Interrupt



Safety Measures

- The robot has ultrasonic sensors which can prevent it from colliding with obstacles.
- It obtains a point cloud from the 3D RGB sensors which can prevent itself from colliding with humans in a certain range.
- It has a kill switch; located at the back of the robot which can be used in emergency cases.
- The velocity at which it is searching is being monitored so that it can not bump into objects.

Modular design and future scope

- The entire product has been designed to be modular.
- Parts can be swapped out for separate functions and future designs. The Learning software itself is independent of hardware specifications.
- Thanks to the distributed nature of ROS, changes can be made quickly and economically.
- This also has the added benefit of massive online support from the ROS community.
- Learning software can be used for robots in other environments.
- Can make variants for warehouses, hospitals, supermarkets etc. with minimal investment in R&D.

Ideas for the future

- Learn to anticipate groups of objects based on patterns of requests.
- Learn profiles for specific users.
- Over time, learn shortest paths about the map.
- Release modified versions for robots to be used in different environments.
- Release soft object handling add-on.
- Learn behaviors over time to assist in daily routines.

Revenue Model

- Lease robots out like a butler service.
- The company takes responsibility of maintaining, calibrating and installing the system.
- Offer subscription on an annual basis.
- Allows us to issue updates to software and hardware relatively easily.
- Allows updates to datasets, and remote training for the learning systems.

Budget

Hardware Tools	Functionality	Price
Dexter	Robotic Arm	$2000
3 Wheel omniwheel drive	Drive for Fetch	$850
RasberryPi 3	Onboard Controller	$50
Intel RealSense	Camera, Depth, Microphone	$110
7“ LCD with multitouch	Display	$125
Adafruit stereo speaker	Speaker	$15
Jetson TX2	Machine Learning Controller	$600
	Miscellaneous Components	$150
	Total Cost	**$3900**

References

- Drive- https://ozrobotics.com/shop/omni-wheel-arduino-robotics-car/?gclid=Cj0KCQiAgZTRBRDmARIsAJvVWAsKc5hzKSS5kCbn26JfHBGd5O4862AOJybf-r6Q9-TWnAItq6q_87waAkSMEALw_wcB
- Arm Camera- https://www.logitech.com/en-us/product/hd-webcam-c310
- Arm- www.7bot.cc
- ROS- http://wiki.ros.org/
- Training sets- http://wiki.ros.org/household%20objects