Robotics Project

Navigation and Localization

F. Jampy[1] C. Prabhu Kumar[2] P. Sai Krishna[3]

Supervisors: S.Ralph[1], D.Cedric[2], D.Fofi[3]
[1]Department of Computer Vision, Le2i
University of Burgundy

Masters in Computer Vision, 2014

Outline

1. Motivation
   - Goals
2. Methodology
   - Navigation in a blank map
   - Navigation in a known map
   - Navigation with obstacle avoidance
   - Target Detection
3. Results
4. Conclusion and Future work

Outline

Our Goals

Navigation in a blank map

- Navigation with one input coordinate with starting pose A and Target pose B.

Navigation in a known map

- Map built with Teleoperation.

Our Goals

Navigation in a blank map

- Navigation with one input coordinate with starting pose A and Target pose B.

Navigation in a known map

- Map built with Teleoperation.

Our Goals

Navigation in a map with Obstacle avoidance

- Navigation with known map and unknown obstacles.

Target Detection

- Visual Servoing.

Our Goals

Navigation in a map with Obstacle avoidance

- Navigation with known map and unknown obstacles.

Target Detection

- Visual Servoing.

Our Goals

Navigation in a map with Obstacle avoidance

- Navigation with known map and unknown obstacles.

Target Detection

- Visual Servoing.

Our Goals

Navigation in a map with Obstacle avoidance

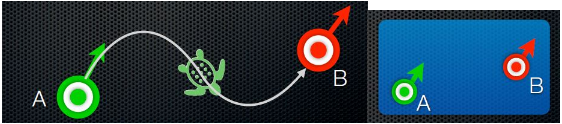
- Navigation with known map and unknown obstacles.

Target Detection

- Visual Servoing.

Outline

Goal 1: Navigation in a blank map



Goal 1: Navigation in a blank map

- Start pose and Goal pose should be same.

Path Planners

- Move_base (Global and Local Planner) and AMCL (Adaptive Monto Carlo Localization) are used.
- AMCL - Laser information and map of the environment are needed.

Goal 1: Navigation in a blank map

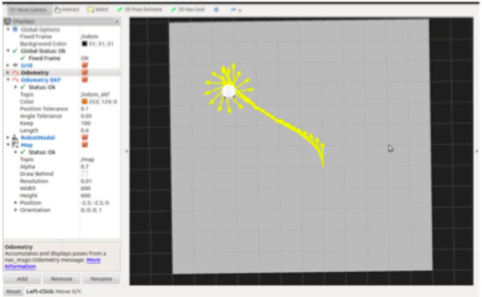
- Start pose and Goal pose should be same.

Path Planners

- Move_base (Global and Local Planner) and AMCL (Adaptive Monto Carlo Localization) are used.
- AMCL - Laser information and map of the environment are needed.

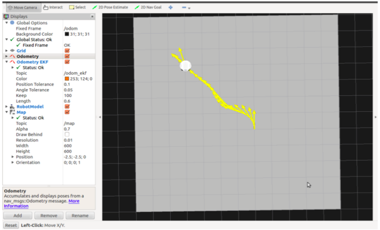
RVIZ Results

using Move_base path planning



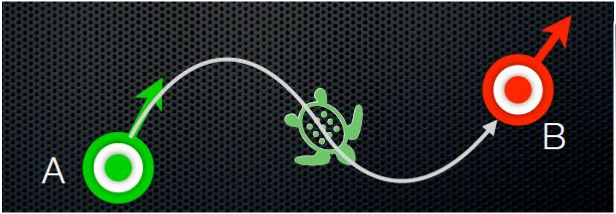
Fake TurtleBot

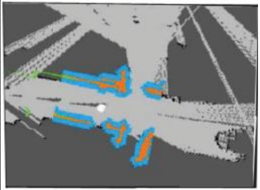
RVIZ Results
using AMCL



Outline

Goal 2: Navigation in a known map





Goal 2: Navigation in a known map.

- Build the Map.

Available Packages in ROS

- Keyboard Teleopertation or joystick.
- Exploration package.
- Interaction marker.

Goal 2: Navigation in a known map.

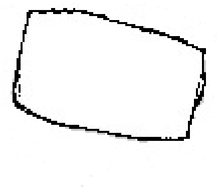
- Build the Map.

Available Packages in ROS

- Keyboard Teleopertation or joystick.
- Exploration package.
- Interaction marker.

Goal 2: Navigation in a known map.

Map built by Teleoperation keys or keyboard teleoperation



Goal 2: Navigation in a known map.

- In order to navigate in a map,
  1. AMCL and Move_base packages are used.
  2. Localize the turtlebot in a map.

Move_base Path Planner

- Based on both global and local planner.
  1. First draws the global planner to reach goal.
  2. Local planner will follow global path. .

AMCL

- Implements the adaptive monte carlo localization approach
  1. which uses a particle filter to track the pose of the robot against the known map.

Goal 2: Navigation in a known map.

- In order to navigate in a map,
  1. AMCL and Move_base packages are used.
  2. Localize the turtlebot in a map.

Move_base Path Planner

- Based on both global and local planner.
  1. First draws the global planner to reach goal.
  2. Local planner will follow global path. .

AMCL

- Implements the adaptive monte carlo localization approach
  1. which uses a particle filter to track the pose of the robot against the known map.

Goal 2: Navigation in a known map.

- In order to navigate in a map,
  1. AMCL and Move_base packages are used.
  2. Localize the turtlebot in a map.

Move_base Path Planner

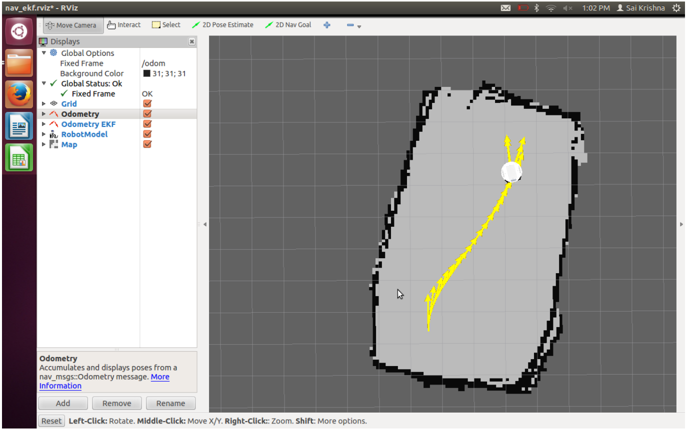
- Based on both global and local planner.
  1. First draws the global planner to reach goal.
  2. Local planner will follow global path. .

AMCL

- Implements the adaptive monte carlo localization approach
  1. which uses a particle filter to track the pose of the robot against the known map.

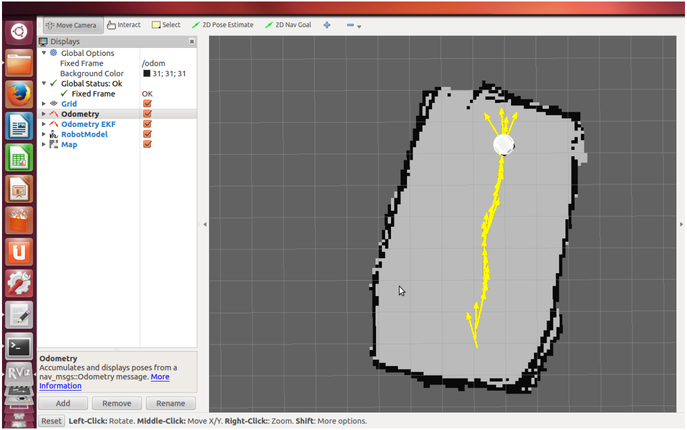
Navigation in a known Map

Move_base without obstacles in a known map in rviz



Goal 2: Navigation in a known map.

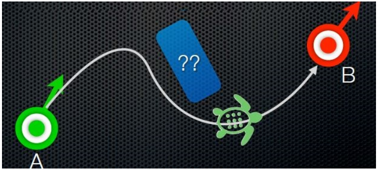
Amcl navigation without obstacles in a known map in rviz.



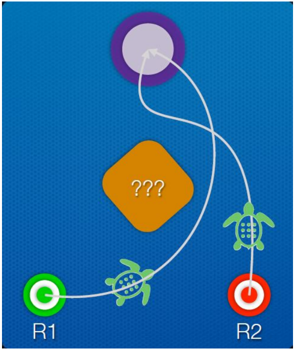
Outline

1. Motivation
   - Goals
2. Methodology
   - Navigation in a blank map
   - Navigation in a known map
   - Navigation with obstacle avoidance
   - Target Detection
3. Results
4. Conclusion and Future work

Goal 3: Navigation with obstacle avoidance



Goal 3: Navigation with obstacle avoidance



Goal 3: Navigation with obstacle avoidance.

- Turtlebot avoids the obstacles while seen on its way(while reaching) the target using any of the path planning packages available in ROS.
- Move_base and amcl are good in avoiding obstacles.
- These overlays are created using the Pause style.

Goal 3: Navigation with obstacle avoidance.

- Turtlebot avoids the obstacles while seen on its way(while reaching) the target using any of the path planning packages available in ROS.
- Move_base and amcl are good in avoiding obstacles.
- These overlays are created using the Pause style.

Goal 3: Navigation with obstacle avoidance.

- Turtlebot avoids the obstacles while seen on its way(while reaching) the target using any of the path planning packages available in ROS.
- Move_base and amcl are good in avoiding obstacles.
- These overlays are created using the Pause style.

Demonstration of Move_base.

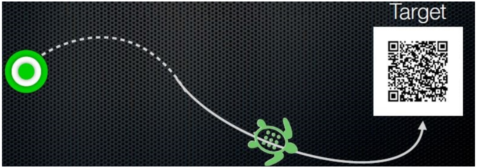
- Move_base is combined with global and local path planner.
- When goal is given, it plans global path.
- When finds the obstacle, local planner will avoid the obstacles and reach the goal.

Demonstration of Amcl.

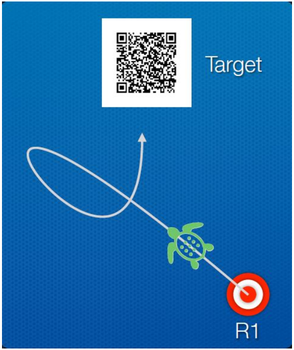
1. Localize the turtlebot in a map. Once the target is given for the turtlebot.
2. Amcl is a probabilistic localization system for a robot moving in 2D.
3. It implements the adaptive Monte Carlo localization approach, which uses a particle filter to track the pose of a robot against a known map.

Outline

Goal 4: Target Detection.



Goal 4: Target Detection.



Goal 4: Target Detection.

Algorithm

1. Step 1: Go from Starting Pose A to Target pose, B.
2. Step 2: Rotate 360º.
3. Step 3: While rotating, if you see the QR code, stop and then goto step6.
4. Step 4: If QR code is not seen, move to the position 'C'.
5. Step 5: Start moving in a square (C,D,E,F). While moving in a square, if QR code seen, then stop.
6. Step 6: Get the position and the orientation values of QR code.
7. Step 7:Give these values to the robot an dit will go and stand infront of the QR code.

Goal 4: Target Detection.

VISP(Visual Servoing Platform)

- To detect the target we tried with different packages like Visp, cmvision & ar_recog.
- First we tried with Visp package for detecting the target and to move towards the target.
- The good thing in visp is we will get the image coordinate i.e. position and orientation with reference to world.

Drawback of VISP

- But the detection is limited. If the distance between turtlebot and target is greater than 20cm we cannot detect the target.
- With this drawback visp is not much helpful in accomplishing the task.

Goal 4: Target Detection.

VISP(Visual Servoing Platform)

- To detect the target we tried with different packages like Visp, cmvision & ar_recog.
- First we tried with Visp package for detecting the target and to move towards the target.
- The good thing in visp is we will get the image coordinate i.e. position and orientation with reference to world.

Drawback of VISP

- But the detection is limited. If the distance between turtlebot and target is greater than 20cm we cannot detect the target.
- With this drawback visp is not much helpful in accomplishing the task.

Blob Finder.

- We tried with blob finder which is available in cmvision package. This could be helpful but too much false positives.

Drawback

- This is package is very sensitive to light conditions in the environment.
- Blob finder image.

Blob Finder.

- We tried with blob finder which is available in cmvision package. This could be helpful but too much false positives.

Drawback

- This is package is very sensitive to light conditions in the environment.
- Blob finder image.

Goal 4: Target Detection

AR_Recognition

- To do robust detection of the target we use ar_recog package- very robust to detect the QR code.
- With this we can able to detect the target from long distance which is up to 2.5m. Not only the distance it also give position and corners of the image.
- With this we are able to detect the target and make the turtlebot to move towards the target.

Goal 4: Target Detection

AR_Recognition

- To do robust detection of the target we use ar_recog package-very robust to detect the QR code.
- With this we can able to detect the target from long distance which is up to 2.5m. Not only the distance it also give position and corners of the image.
- With this we are able to detect the target and make the turtlebot to move towards the target.

Goal 4: Target Detection

AR_Recognition

- To do robust detection of the target we use ar_recog package-very robust to detect the QR code.
- With this we can able to detect the target from long distance which is up to 2.5m. Not only the distance it also give position and corners of the image.
- With this we are able to detect the target and make the turtlebot to move towards the target.

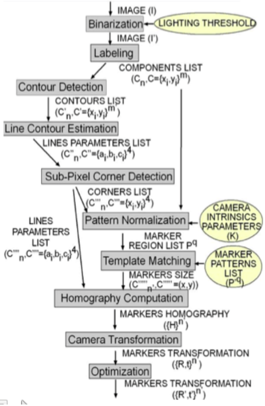
Goal 4: AR_Recognition.

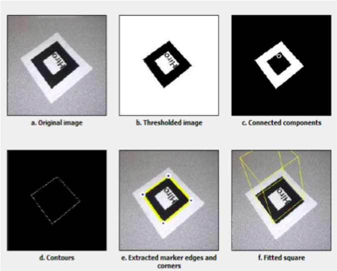
- ar_recog idea ?
- First we obtain the markers of the image from online source.

Goal 4: AR_Recognition.

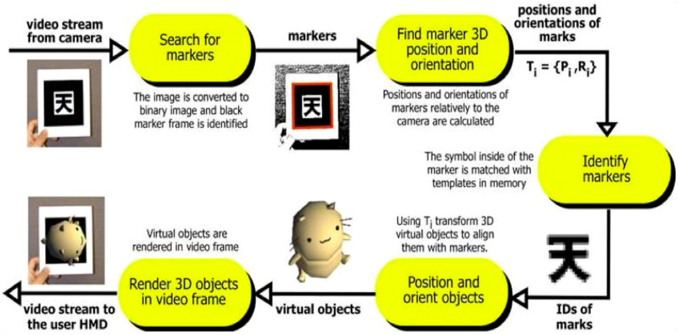
- ar_recog idea ?
- First we obtain the markers of the image from online source.

Goal 4: AR_Recognition.



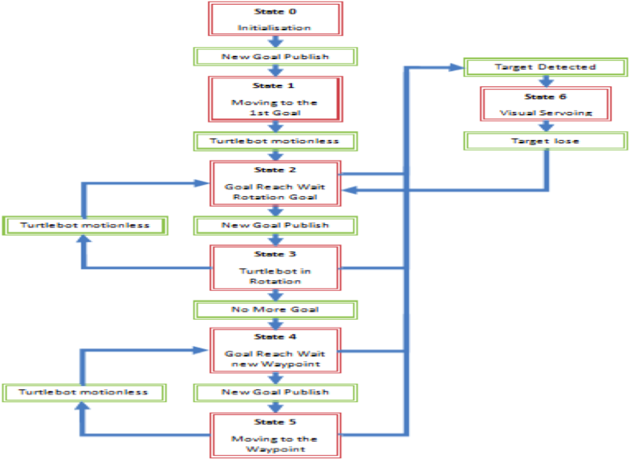


Goal 4: Working of AR_Recognition.



1. To achieve the final goal , moving from zone 1 to zone 2 and to identify the target we take advantage of state machine.
2. For this we gave altogether 6 states for turtlebot to detect the target and move towards the goal.
3. Each state represent operation in each step to achieve the target.

State Machine Execution Picture/ Flow chart



References

R.Patrick Gabriel
ROS by examples.
volume 1.

http://wiki.ros.org/gmapping.
Wikipedia.

http://wiki.ros.org/visp.
Wikipedia.

http://wiki.ros.org/cmvision.
Wikipedia.

http://wiki.ros.org/ar_recog.
Wikipedia.