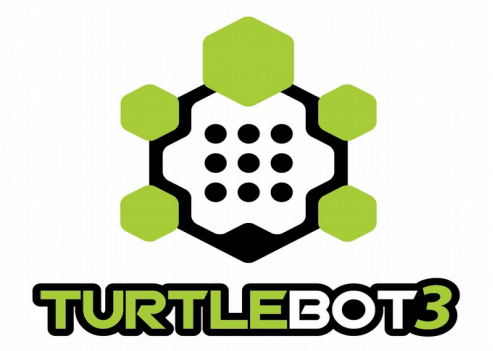


Master in Computer Vision

UNIVERSITE DE BOURGOGNE

Robotics Project

Technical Report

Supervisor : Ph.D. Ralph SEULIN

Student: Martin Emile

Final Report for the course titled “Robotics Project”

Centre Universitaire Condorcet - UB, Le Creusot

December 6, 2020

Contents*

0-Abstract

Demonstration of tutlebot3 robot control and simulation is achieved through different completed tasks, which in order to be solved, a certain skills related to ROS basics in programming and architecture understanding should be mastered as described in the initial report, on top of that, Our simulated ROS environment provided by The-Construct website is exploited with the different provided ROS online courses as well as technical mentoring. As a result, our implementation been able to achieve the 4 requested tasks. In this report, a detailed guideline about the implementation will be presented in order for the reader to be able to implement it him/her self.

1- Introduction

TurtleBot3 is a one low-cost, personal robot with open-source software, that can perform different interesting goals. According to [1], TurtleBot3 is a small, affordable, programmable, ROS-based mobile robot for use in education, research, hobby, and product prototyping. The goal of TurtleBot3 is to dramatically reduce the size of the platform and lower the price without having to sacrifice its functionality and quality, while at the same time offering expand-ability. The TurtleBot3's core technology is SLAM, Navigation and Manipulation, making it suitable for home service robots. The TurtleBot can run SLAM(simultaneous localization and mapping) algorithms to build a map and can drive around the environment. Also, it can be controlled remotely from a laptop, joy-pad or Android-based smart phone as described in [2]. The TurtleBot can also follow a person's legs as they walk in a room. Our goal is to implement 4 different robotics tasks on Turtlebot3 robot, that is based on mainly on knowing how to use ROS to manipulate the robot, simulate an environment, Path planning and localization and way-points following.

2-Demo Videos:

https://drive.google.com/drive/folders/191sR8DYO83IFeH6Y3nk76CO7pdj95PTV?usp=sharing

COMPLETED

Robotics Project Implementation

Project Progress pipeline

PROJECT START

1st Revision
-Linux
-Python

2nd Revision
-Ros in 5 days

Publishing
-Simple program to publish into a topic «cmd_val»

Continue
-Ros Navigation in 5 days

Review
-Mastering Turtlebot3 first 2 lessons

Mapping
-Map using slam_gmapping node

Review
-Mastering Turtlebot3 3rd lesson

Localization
-Amcl node from the Navigation Stack

Path Planning
-Use the move_base node from the Navigation Stack

Review
-ROS package called follow_waypoints

Preparation
-Launch Files for follow_waypoints

Test&Debug
-Double check before submition

15/11 16/11 - 18/11 20/11 - 25/11 28/11 - 2/12 3/12 - 07/12

Task 1

Publish into cmd_val

Task 2

Map the environemnt

Task 3

Path Planning

Task 4

Way points following

4-Problem approach

For task achieving, you can choose to either download ROS on your Linux system with all the requirements that comes before that, or you can choose to implement your project on The-Construct platform that has all the necessary ROS packages already installed.

After that, Get ready to review your Linux basic shell commands, for example,you should learn, How to navigate through a Linux file system. How to interact with a Linux file system, How to edit files using the Shell (vi editor). Manage access to files (Permissions).Create simple Linux programs (Bash Scripts). Manage execution of Linux programs (Processes).How to connect to the remote computer of a robot (ssh).

Moreover, If you are rusty on your basic python syntax, get ready to brush-up some basics, for instance, you can learn, How to store data into Variables. How to operate with the data in the Variables. How to change behavior based on Conditions. How to create Functions that can be called from other places of the code. How to encapsulate the code into Classes so you can have clean and robust code. As a tutorial, The-construct offers two free courses regarding Linux basics and python basics for ROS, I highly recommend to check them.

As for ROS basics, I recommend you choose either a reliable tutorial online or you can buy a book like [3], and stick to them all the way while developing along side your chosen tutorial. It is essential to cover, how the ROS system works regarding different nodes and how they interest with each other using topics, how to publish into a topic, how to get information about different ROS messages. How to mapping works, how to go to goal..etc.

After that, you will be ready to go and tackle the following 4 tasks:

1-Script that moves the robot around with simple /cmd_vel publishing.

2-Map of the environment and. Setup, the launch to be able to localize the Turtlebot3 robot.

3-Publish a goal to move_base and Turtlebot3 can reach that goal without colliding with obstacles.
4-Navigate within the environment following a set of waypoints.

5-Basic steps for Task 1:

1-Open a Linux shell window

2- Enter " $ rostopic list -grep /cmd_vel" to make sure the topic exist

3-Enter "$ rosmsg show geometry_msgs / Twist" to see the structure of the messge you want to send through the topic

4- Finally, Enter "$rostopic pub / cmd_vel geometry_msgs / Twist - '[2.0, 0.0, 0.0]' '[0.0, 0.0, 0.0]'" to see the robot moving in the x direction.

6-Basic steps for Task 2:

1- Extract my implemented package that comes along with this report

2- Place it in your Catkin_ws/src file

3- Before we start launching launch files, I would recommend to take a look at [4] for some information about the g_mapping node

4- Launch Rviz to visualize the mapping process

5- Launch the "start_mapping.launch" launch file

6- Move the robot using the keyboard launch file

7- Visualize the mapping while being built

8- Make a directory to place the map files

9- Save the Map

10- For the localization part, launch "start_localization.launch" file

11- Add a PoseArray into your Rviz that subscribe to /ParticleCloud

12- Move the robot using the Keyboard

13- Observe the red arrows getting denser on the location of the robot

7-Basic steps for Task 3:

1- Extract my implemented package that comes along with this report

2- Place it in your Catkin_ws/src file

3- For the path planning and navigation, launch the " start_navigation.launch" launch file

4- Launch Rviz to visualize the navigation process

5- Using Rviz 2D naviagte button, send a position in the environment

6- Kindly, note, the obstcale avoidance is already taken into account because we already mapped the environment with all the existing obstacles

7- Visualize the navigation process

13- Observe the path taken by the robot

8-Basic steps for Task 4:

First method, choosing way points manually

1- Extract my implemented package that comes along with this report

2- Place it in your Catkin_ws/src file

3- Clone this library https://github.com/danielsnider/follow_waypoints

4- Launch the “ start_navigation.launch” launch file that in the t3_navigation package

5- Launch the “follow_waypoint.launch” launch file that is in the cloned package

6- Start Rviz for visualizing

7- Add A PoseArray to your Rviz that subscribe to the topic /way_points

8- Choose your way points using the Rviz 2D Pose Est. button

9- After choosing your way points, publish into the topic /path_ready

10- Observe the robot follow the specified way points

Second method, choosing way points using a python script

1- Extract my implemented package that comes along with this report

2- Place it in your Catkin_ws/src file

3-Launch the “ start_navigation.launch” launch file that in the t3_navigation package

5- Launch the “start_follow_waypoint.launch” launch file that is included in my package

6- Start Rviz for visualizing or Gazebo

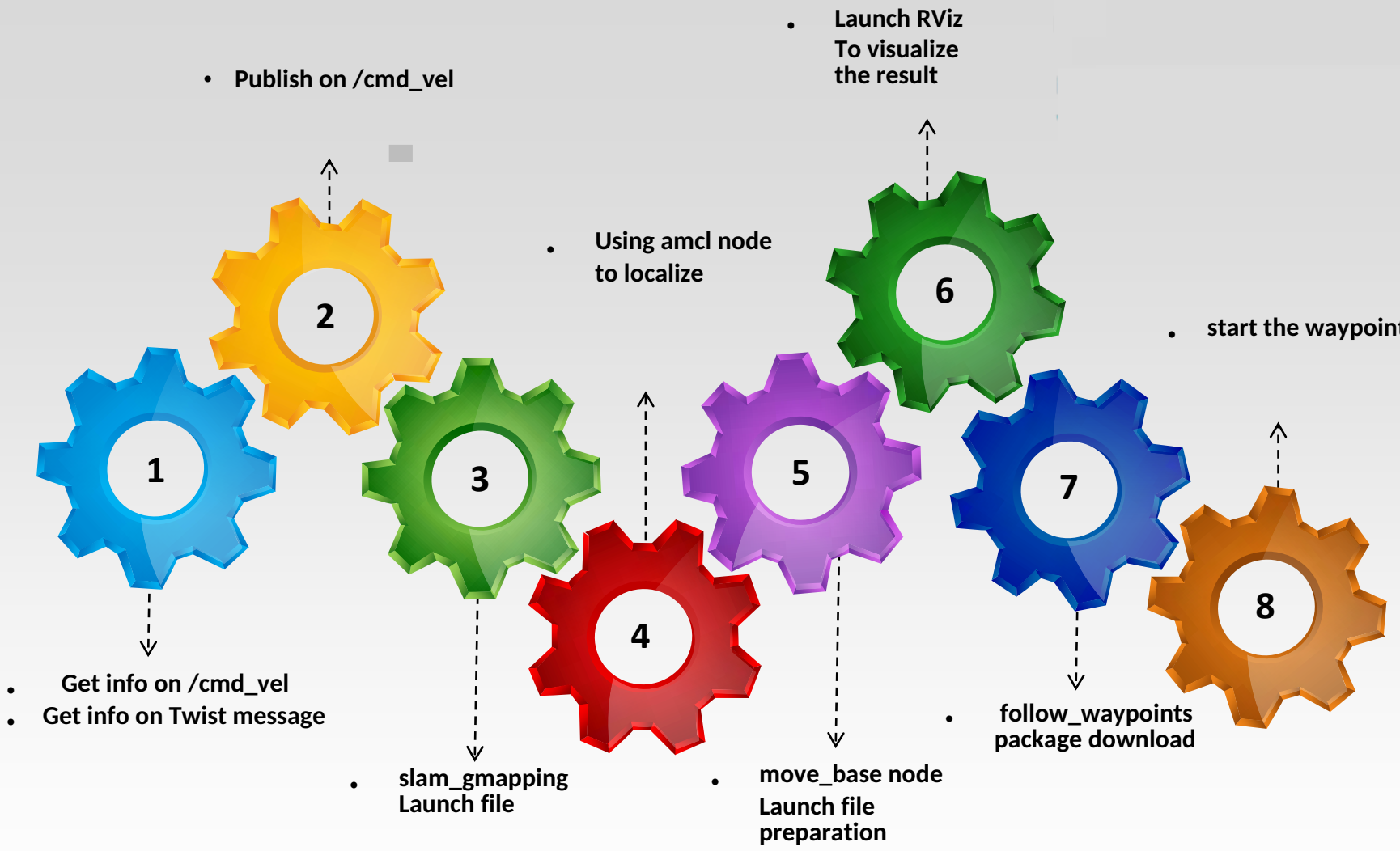
7- Add A PoseArray to your Rviz that subscribe to the topic /way_points

8- Specifiy your way_points in the python file titled “waypoint_publisher.py”

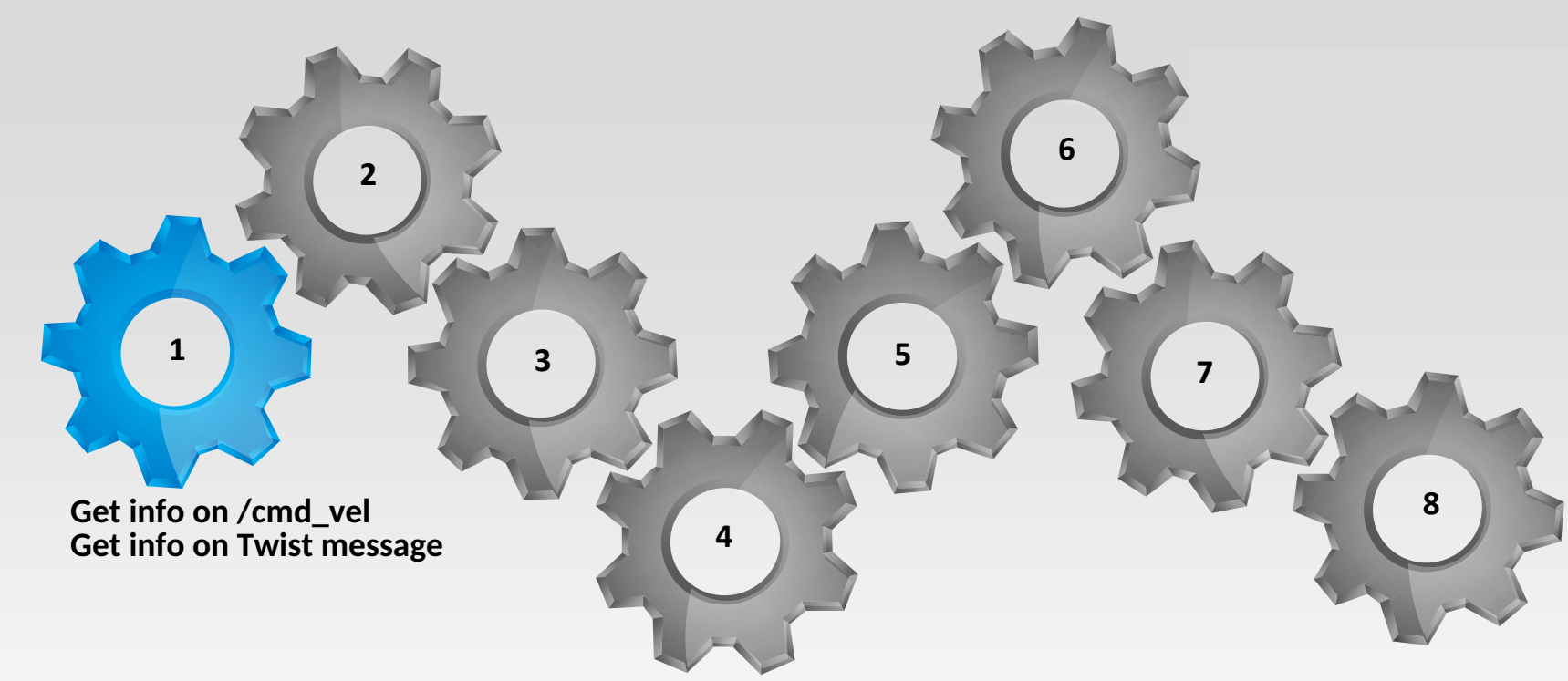
9- After choosing your way points, publish into the topic /path_ready using the python file “waypoint_publisher.py” so just execute this file

10- Observe the robot follow the specified way points

Project Process – 8 Stages to the 4 Robotics Tasks



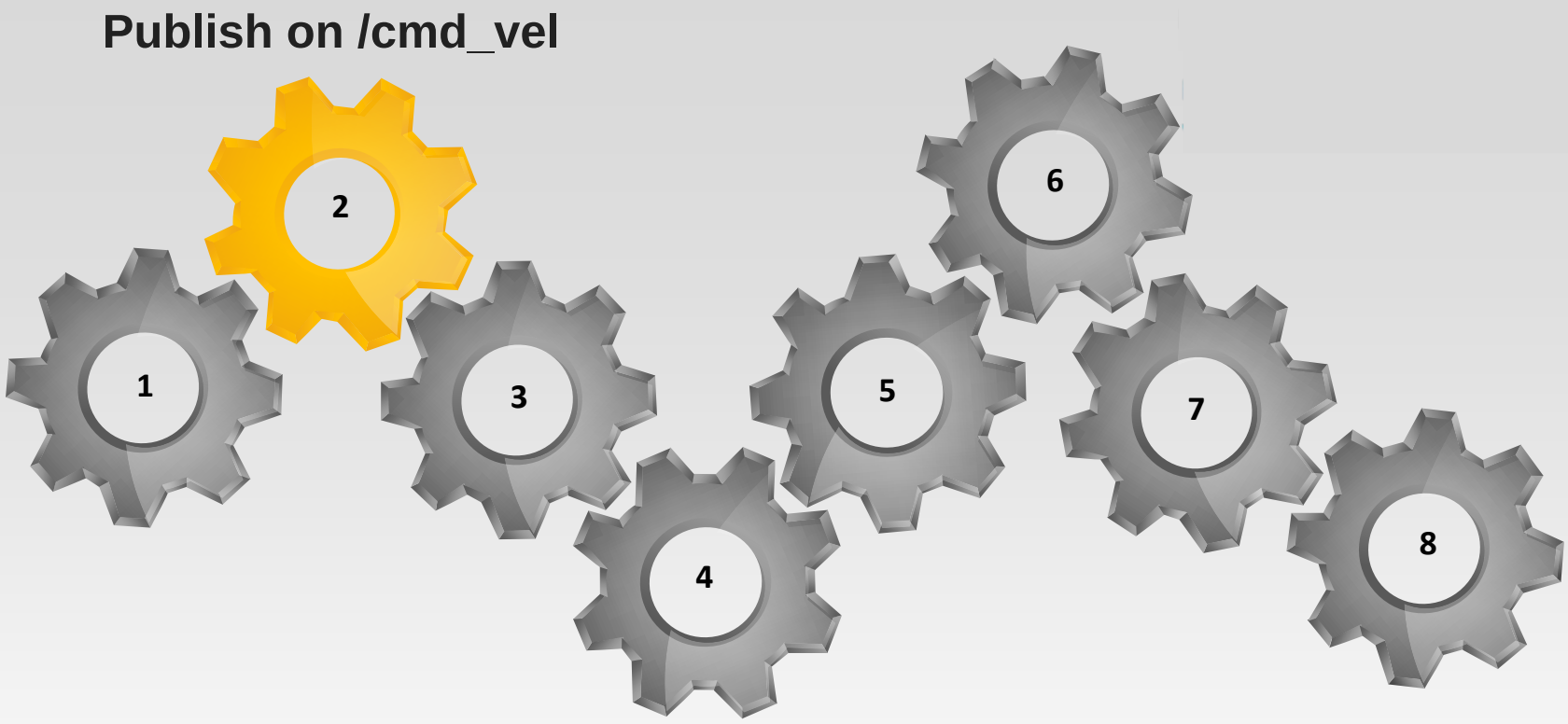
Project Process – 1st Stage



1st Stage

Create a package to place the necessary fills
Get info with “rostopic info /cmd_vel”
, after this,get info of geometry_msgs/Twist message. So, this is the type of message that we will need to send to the topic in order to control the velocities.

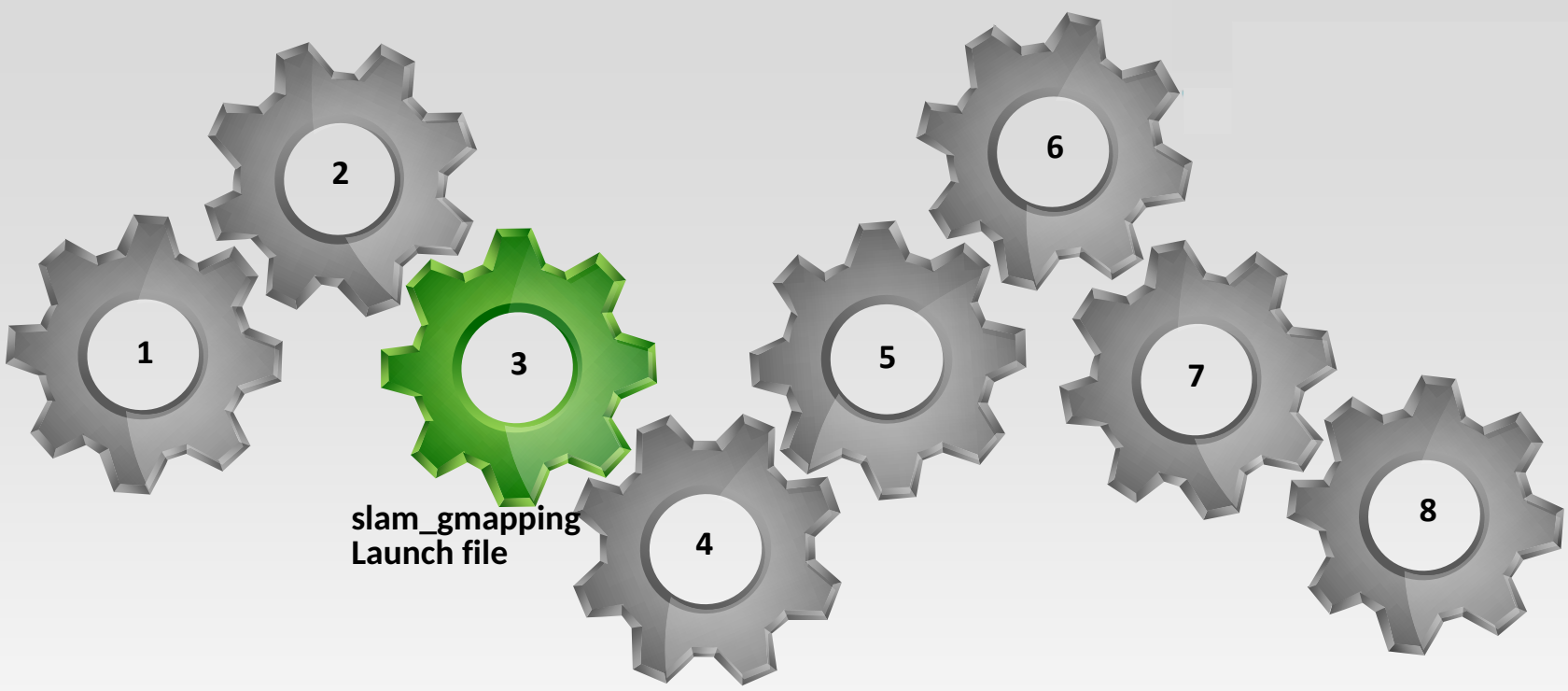
Project Process – 2nd Stage



2nd stage

- Get the structure we use “rostopic pub /cmd_vel [TAB][TAB]”
- Enter the proper Velocities Linear in X,Y and angular in Z.

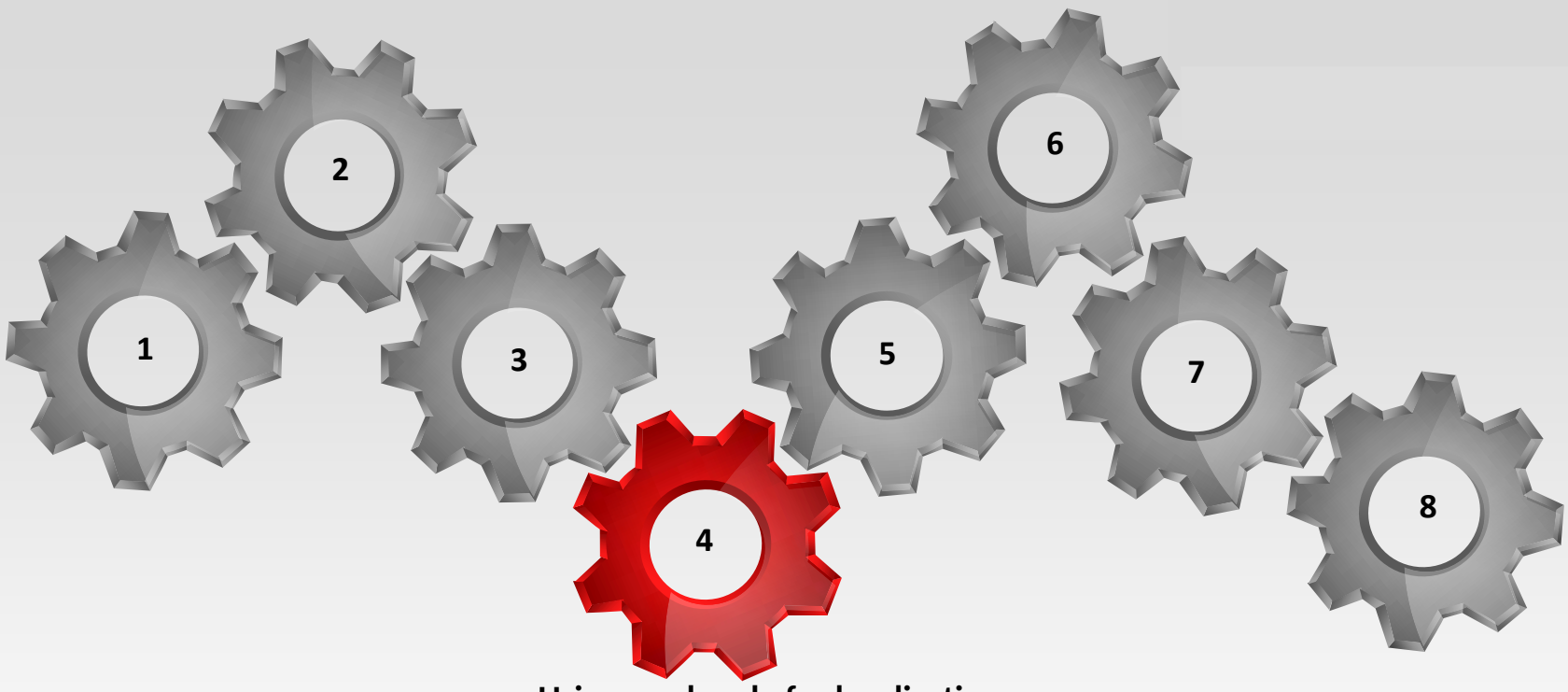
project process – 3rd Stage



3rd Stage

create a launch file in order to start our slam_gmapping node!Using the Navigation Course” ROS in 5 days” Mapping notebook in order to tailor the launch file parameter

project process – 4th Stage

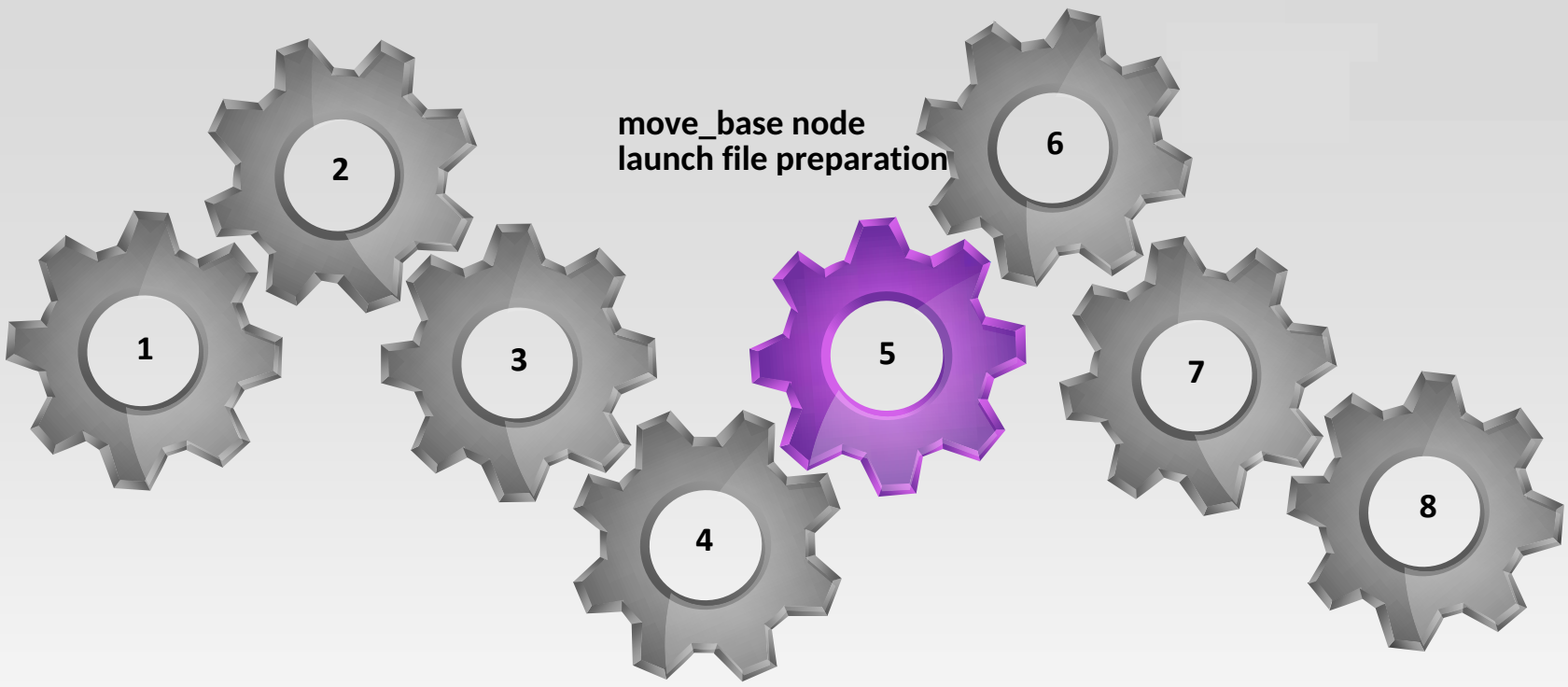


Using amcl node for localization

4th Stage

After Running the map_saver node, which allows us to access the map data from a ROS Service, and save it into a file. we need to Localize the robot into that map, we are going to use the amcl nodefrom the Navigation Stack , in addition, we will Launch RViz in order to be able to visualize the localization process

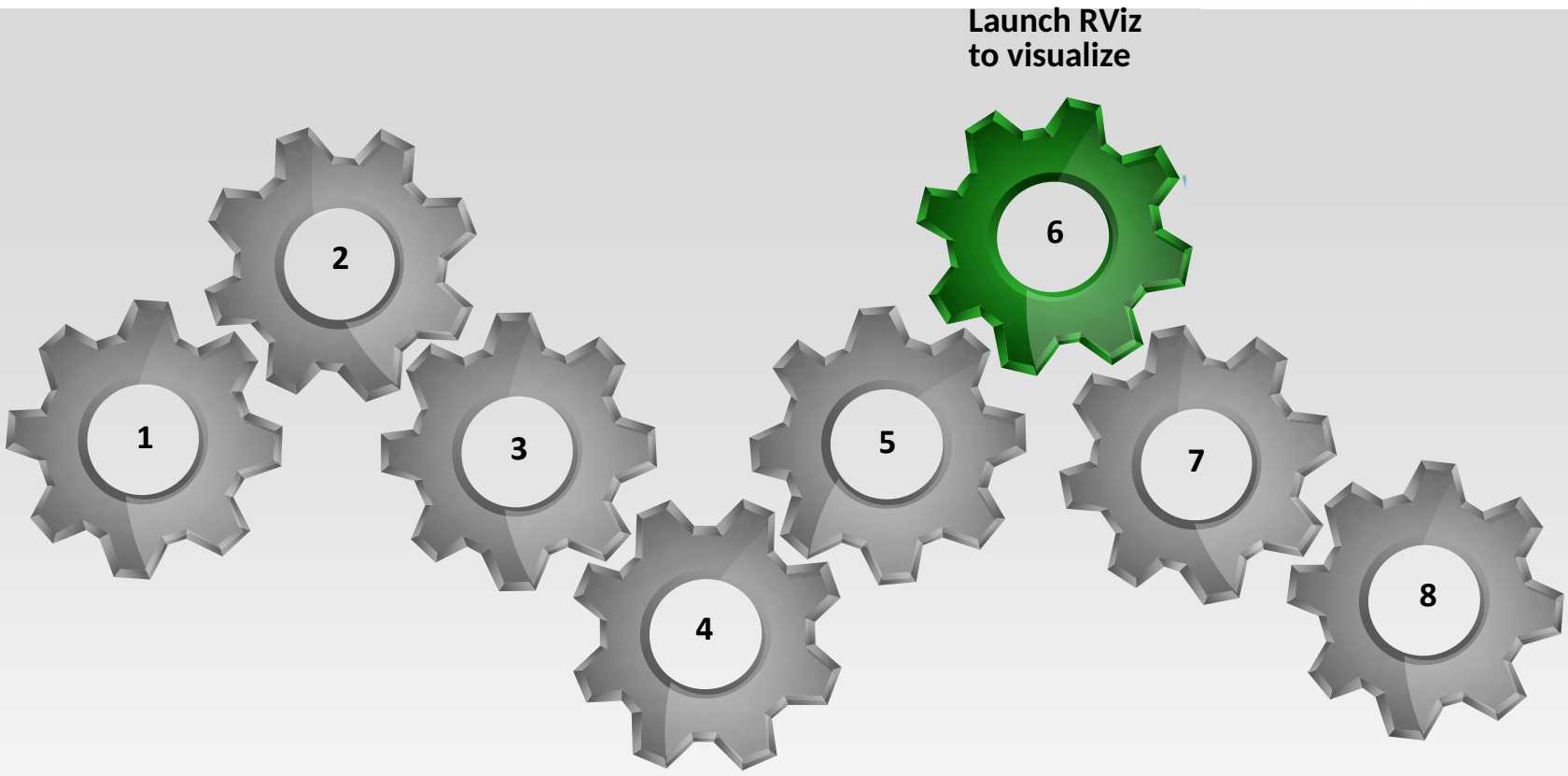
project process – 5th Stage



5th Stage

Create a package to create a new launch file in order to start the move_base node. Using the Course named “ ROS Navigation in 5 days” with the notebook dedicated to path planning change the parameter inside the launch file for your customized task adjust all the parameter required by the move_base node.

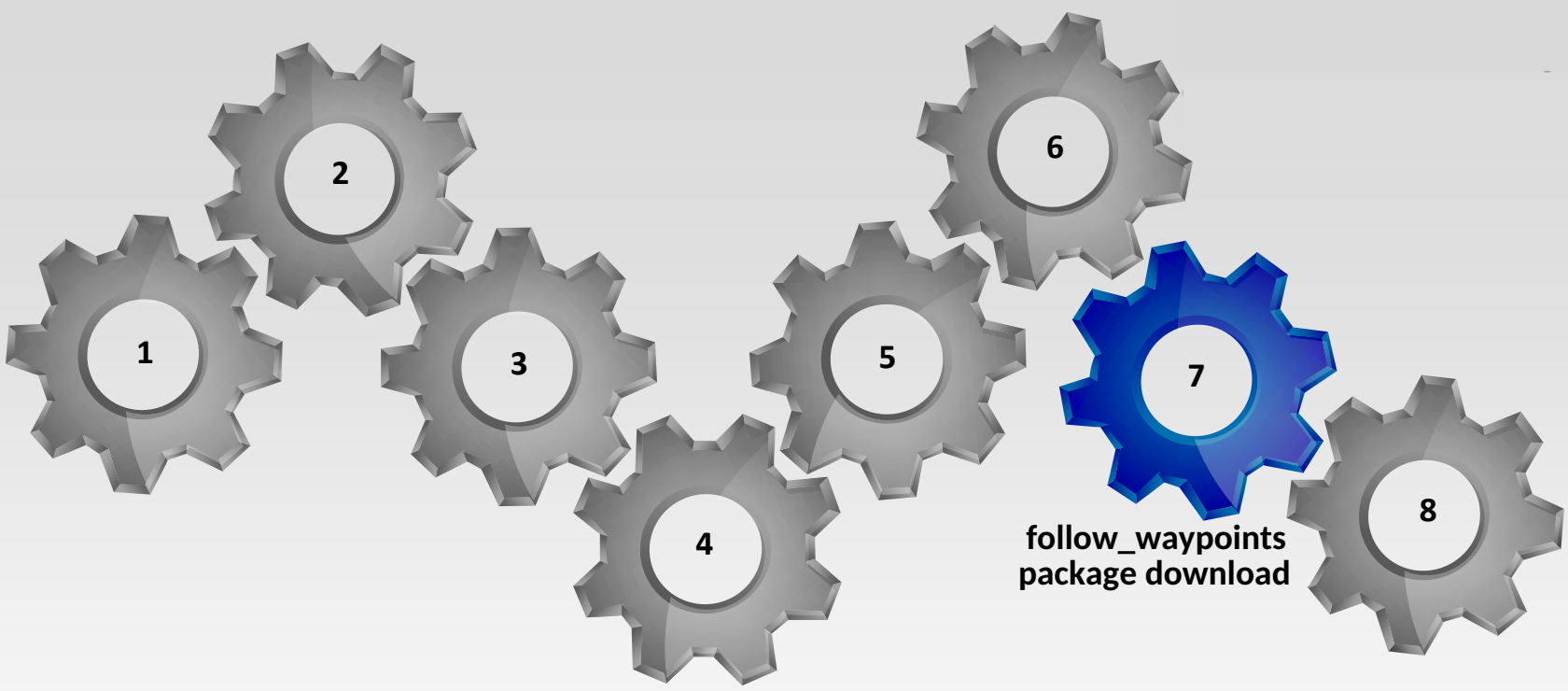
project process – 6th Stage



6th Stage

ROS graphical interface that allows you to visualize a lot of information, using plugins for many kinds of available topics.Launch RViz in order to be able to visualize the Path Planning process

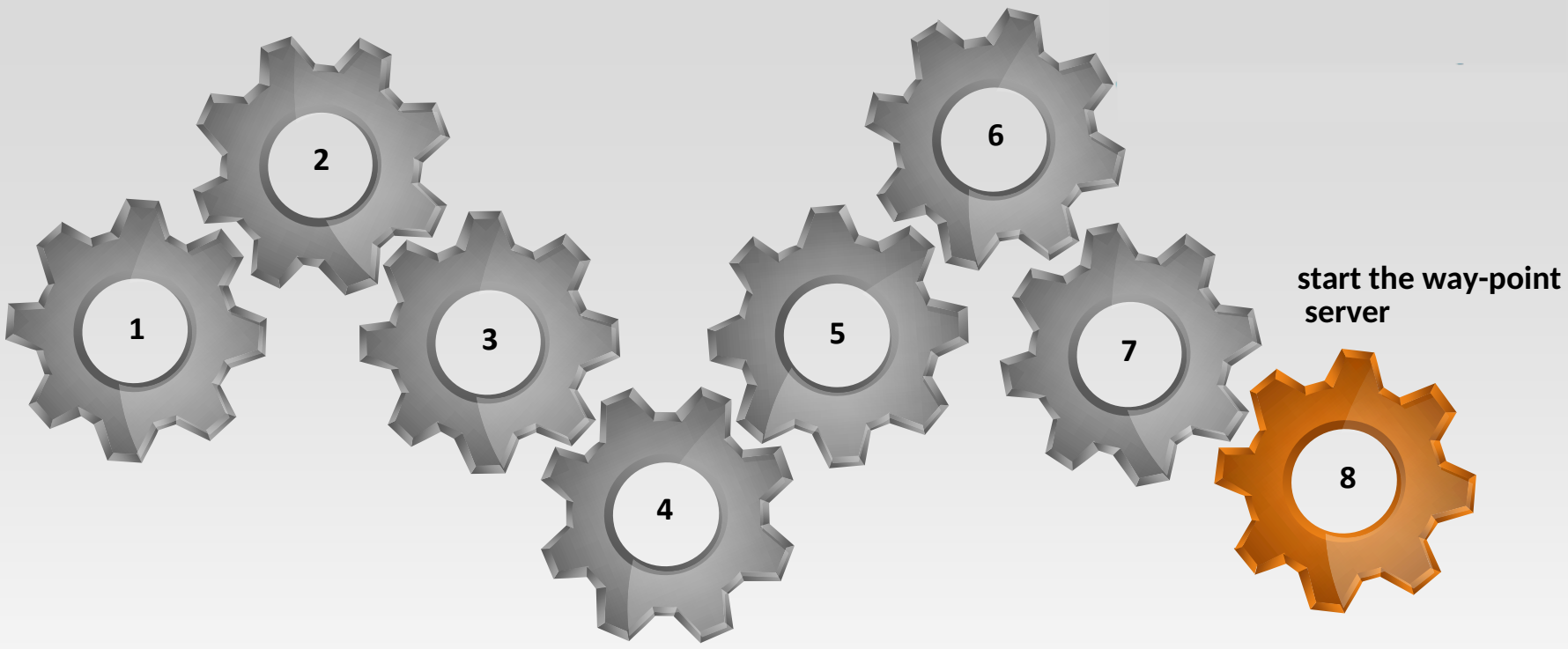
project process – 7th Stage



7th Stage

Download the follow_waypoints package to the construct workspace, then start the launch file in Task 3, if I did not have it running already, then start the waypoint server, in order to listen to publications into the topic /initialpose and store those poses until its instructed to send them to move_base node to be executed

project process – 8th Stage



8th Stage

Start RVIZ, and add a PoseArray element, subscribe to the topic /waypoints, and change its name to WayPoints. then, select the PoseEstimate and set it where you want it. Because this will move the estimated pose of the robot
Publish in the topic /path_ready to start sending waypoints to the movebase node.

10-Conclusion:

To conclude, a synthesis of the necessary instructions required in order to be able to achieve the tasks using Turtlebot3 has been presented alongside the implemented ROS package, on top of that, expression of different ROS learned concepts and techniques has been synthesized in order to be used in the different aspects during the project implementation, in addition, a work plan for the main project pipeline has been presented in a graphical context.

11-References:

[1] https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/

[2] https://www.roscomponents.com/en/mobile-robots/214-turtlebot-3.html#/courses-no

[3] Book "ROS Robotics Projects"

[4] http://wiki.ros.org/gmapping

[5] Course titled “Mastering with ROS: Turtlebot3”

[6] Course titled “ROS Navigation in 5 days”

[7] http://wiki.ros.org/ROS/Tutorials/ExaminingPublisherSubscriber

[8] http://wiki.ros.org/navigation