Task 5 - Original Configuration

1 Problem Statement

Aim of this task to implement the entire Survey and Rescue Theme's problem statement.

2 LED Configuration:

Cell Location	LED module No
E5	1
D4	2
B3	3
C5	4
A6	5
A2	6
C3	7
E2	8
E4	BASE

3 Procedure

1. Update to the latest survey_and_rescue package from Github.
2. **Please make the following changes in the new detect_beacon.launch file available on Survey and Rescue's repository .**
   - **Line 3**: change *duration* param from *25* seconds to *200* seconds (The run will still be of 3 minutes run as per Rulebook)
   - **Line 9**: Make *start_on_base* param *True*. To start the LEDs once the Drone is hovering over the base (at the base coordinate from json file).
   - **Line 10**: Make *start_with_countdown* param *False*. This was done to make sure that *rosserial* node has sufficient time to initialised. However, since we are initialising *rosserial* node in *prerequisites.launch* file, the purpose of this parameter is not applicable now.
   - **Line 15**: Change *continuous_hovering* param to *False*. This param decides to factor the hovering nature of the drone either continuously (as per mentioned in RuleBook) or cumulatively (easier, as decided in Task 4.2). Making it False ensures it is in cumulative mode.

- **Line 32**: In the *type* param change from *monitor.pyc* to *monitor_org_config.pyc*

3. We've provided you with the LED_Org_Config.tsv (named as LED_Config.tsv in earlier tasks) along with this document.

4. Please read this file through your scheduler to know the *Base* and placement of the LEDs on the cells according to serial number mentioned.

5. For this task we've included the functionality of *LED_Timing.tsv* inside the monitor_org_config.pyc itself to assert uncertainty. The events should only be detected by your *beacon_detection.py* script and decision whether to provide service to that uncertain beacon now or later will depend upon your *scheduler.py* script.

   NOTE: To gain information regarding *Beacons*, only image-processing should be used i.e. by running your *detect_beacon.py* script .The usage of *currentLit* information from the *stats_sr* topic is strictly prohibited.

6. Launch or run all the required nodes. Follow the steps to generate Task 5 output:
   We strongly recommend you to adhere to this methodology so as to maintain uniformity.

   (a) Start roscore.

   (b) launch prerequisites.launch by typing the following command:

   ```
   roslaunch survey_and_rescue prerequisites.launch
   ```

   This will initialise the following node:

   - **USB camera node**
   - **Whycon node**
   - **Rosserial node**

   NOTE: Change the display output of the prerequisites.launch file from /whycon/image_out to /usb_cam/image_rect_color. This is a mandatory requirement for the final task video submission. Refer the updated prerequisite.launch on the survey_and_rescue's repository.

   (c) Next steps would be to:

   - Make sure to place the Drone at the Base location at the beginning itself. This is done to avoid any human intervention in-between the run.
   - Establish connection with the Drone.

     ```
     rosrun edrone_client edroneclient
     ```

   - Start your position hold script.

     ```
     rosrun survey_and_rescue position_hold.py
     ```

   (d) Run your *scheduler.py* script:

   ```
   rosrun survey_and_rescue scheduler.py
   ```

(e) Once, all the previous nodes are up and running, launch detect_beacons.launch file:

```
roslaunch survey_and_rescue detect_beacons.launch
```

This will initialise the following nodes:

- **monitor node:** monitor_org_config.pyc script.
- **sr_beacon_detector node:** your beacon_detection.py script.
- **Rosbag Recording node:** will record a 3 minutes bag file.

We advise you to always record your each run, along with maintaining a log of that run, so at the time of submission you can avoid wrong bag file upload.

```
roslaunch survey_and_rescue detect_beacons.launch record:=True
                    rec_name:=nth_iteration.bag
```

Like mentioned before, monitoring script will only start the LEDs once your drone will holds it's position at the co-ordinates of the base (The same x,y & z mentioned in your json file as well).

7. After, LEDs starts glowing, your *scheduler.py* script should inform the position hold script about cell which it needs to service, refer the Task 4.2 document for more information.

8. At the end of the 3 minutes run. The monitor script (in this case the *monitor_org_config.pyc* script) will publish the following message so as to inform you the end of the run.

   location:"E4"
   info:"END"

   Where E4 is the Base Location for Original Configuration.

4 Expected Output

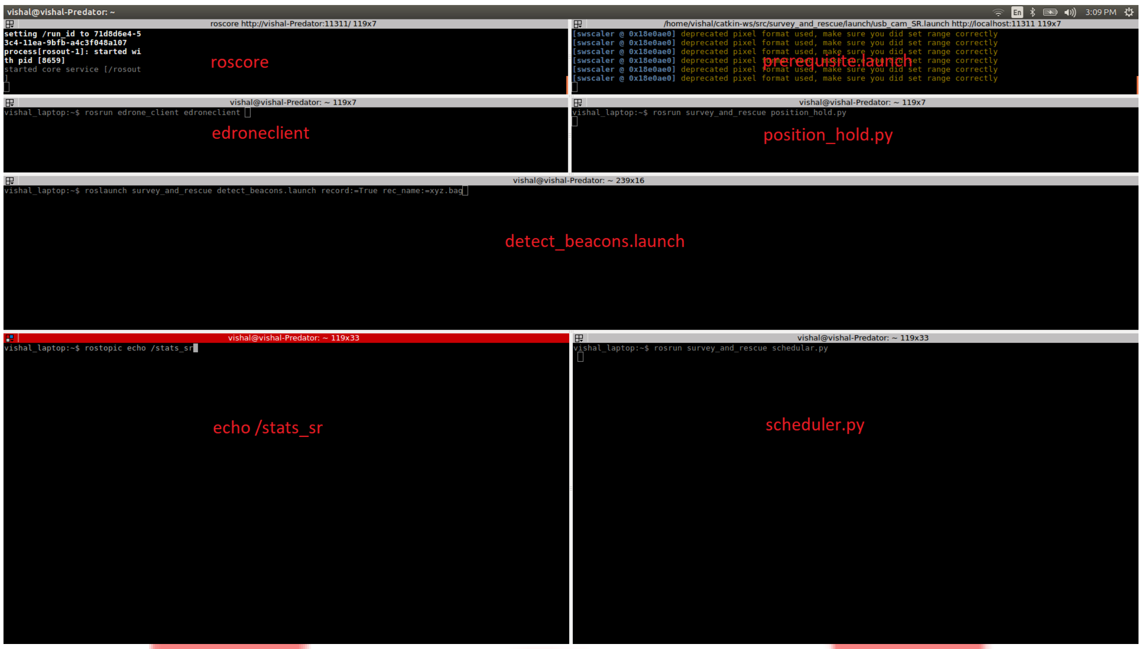
It is extremely important that you adhere to these steps while creating a video for your submission. Not following these step may effect your evaluation.

1. Your screen recording should begin with the Title Slide. As you've been doing so far.

2. Next, please be sure of using the same Terminator format as shown in Figure 1.

   Each terminal has a designated command. These commands are highlighted by red letters.

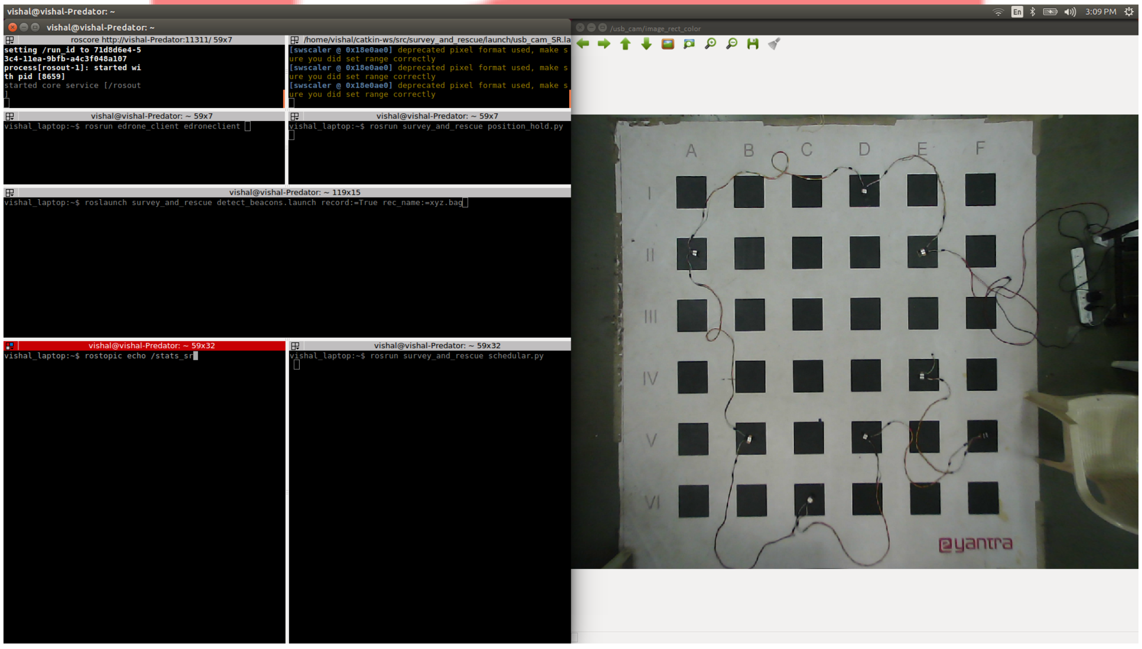
3. Make sure, the `/stats_sr` terminal window is big enough to cover all the line clearly inside this topic.



```
vishal@vishal-Predator: ~                                                        3:09 PM

roscore http://vishal-Predator:11311/ 119x7
setting /run_id to 71d8d6e4-5
3c4-11ea-9bfb-a4c3f048a107
process[rosout-1]: started wi
th pid [8659]
started core service [/rosout
]

/home/vishal/catkin-ws/src/survey_and_rescue/launch/usb_cam_SR.launch http://localhost:11311 119x7
[swscaler @ 0x18e0ae0] deprecated pixel format used, make sure you did set range correctly
[swscaler @ 0x18e0ae0] deprecated pixel format used, make sure you did set range correctly
[swscaler @ 0x18e0ae0] deprecated pixel format used, make sure you did set range correctly
[swscaler @ 0x18e0ae0] deprecated pixel format used, make sure you did set range correctly
[swscaler @ 0x18e0ae0] deprecated pixel format used, make sure you did set range correctly
[swscaler @ 0x18e0ae0] deprecated pixel format used, make sure you did set range correctly

vishal@vishal-Predator: ~ 119x7
vishal_laptop:~$ rosrun edrone_client edroneclient

vishal@vishal-Predator: ~ 119x7
vishal_laptop:~$ rosrun survey_and_rescue position_hold.py

vishal@vishal-Predator: ~ 239x16
vishal_laptop:~$ roslaunch survey_and_rescue detect_beacons.launch record:=True rec_name:=xyz.bag

vishal@vishal-Predator: ~ 119x33
vishal_laptop:~$ rostopic echo /stats_sr

vishal@vishal-Predator: ~ 119x33
vishal_laptop:~$ rosrun survey_and_rescue schedular.py
```

Figure 1: Expected terminal configuration for video

4. Once, you've started your *roscore* and *prerequisite.launch*. Tile (snap) the terminator window on to the left side of the screen and the /usb_cam/image_rect_color window on right side of the screen as shown in Figure 2. After this is done, proceed with your run as mentioned in the procedure.



```
vishal@vishal-Predator: ~                                                        3:09 PM
vishal@vishal-Predator: ~

roscore http://vishal-Predator:11311/ 59x7
setting /run_id to 71d8d6e4-5
3c4-11ea-9bfb-a4c3f048a107
process[rosout-1]: started wi
th pid [8659]
started core service [/rosout
]

/home/vishal/catkin-ws/src/survey_and_rescue/launch/usb_cam_SR.la
[swscaler @ 0x18e0ae0] deprecated pixel format used, make s
ure you did set range correctly
[swscaler @ 0x18e0ae0] deprecated pixel format used, make s
ure you did set range correctly
[swscaler @ 0x18e0ae0] deprecated pixel format used, make s
ure you did set range correctly

vishal@vishal-Predator: ~ 59x7
vishal_laptop:~$ rosrun edrone_client edroneclient

vishal@vishal-Predator: ~ 59x7
vishal_laptop:~$ rosrun survey_and_rescue position_hold.py

vishal@vishal-Predator: ~ 119x15
vishal_laptop:~$ roslaunch survey_and_rescue detect_beacons.launch record:=True rec_name:=xyz.bag

vishal@vishal-Predator: ~ 59x32
vishal_laptop:~$ rostopic echo /stats_sr

vishal@vishal-Predator: ~ 59x32
vishal_laptop:~$ rosrun survey_and_rescue schedular.py
```

Figure 2: Final frame of the Entire Video

5 Points to Remember:

1. Make sure while creating the video or while practicing itself there should be **zero human intervention.**
2. During run, if your drone goes out of arena. Refer section 7.6 of RuleBook regarding repositioning. You can only restart *edroneclient* and *position_hold* node after placing the drone at the Base. Nominate **only a single person to reposition the drone**. Multiple team member should intervene while doing so.
3. **While recording, obstacle like chair (as you can see in Figure 2) or even legs of team member should not be inside the frame.**

6 Submission Instructions

Follow the instructions below to submit your Final Task.

6.1 Bag File:

- Next, when you want to record the bag file for submission, run the following launch file.

```
roslaunch survey_and_rescue detect_beacons.launch record:=True
                        rec_name:=SR_123_5.bag
```

  Where instead of 1234, your team ID should be written.

  WARNING:

  - For recording/creating the bag file, use ONLY detect_beacons launch file. Do not manually record the bag file. This done to maintain uniformity.
  - Like recommended in section 3.4.c, record all your runs and don't forget to rename it before submission.

- Before submitting make sure to verify your bag file, by Check the number of messages, which should be non-zero. By running the following commands.

```
rosbag info SR_123_5.bag
```

- Further verification can be done by playing the rosbag itself. This will also help you know your own score as well.

```
rosbag play SR_123_5.bag
```

Where instead of 123, your team ID should be present.

While echoing `stats_sr` topic in other terminal for verification.

- Make sure your rosbag has recorded the following topics:
  - */stats_sr*
  - */decision_info*
  - */detection_info*
  - */serviced_info*
  - */whycon/poses*

NOTE:
Repeated running of this launch file will overwrite the SR_123_5.bag file, please make sure you have a backup. Alternatively, you can use the argument rec_name param have a custom name while executing the detect_beacons launch file.

6.2 Python Scripts and dependent files:

- You must submit all of your python script developed in this theme uptil now. Basically your entire ***survey_and_rescue/scripts*** folder.
- The following ***scripts*** should be inside the your scripts folder:
  1. The position hold and waypoint navigation script. Rename it to ***position_hold.py*** if not the same already.
  2. The scheduler script. Rename it to ***scheduler.py*** if not the same already.
  3. The Beacon detection script. Rename it to ***beacon_detector.py*** if not the same already.
  4. Cells co-ordinate JSON file. Rename it ***cell_coords.json*** if not the same already.

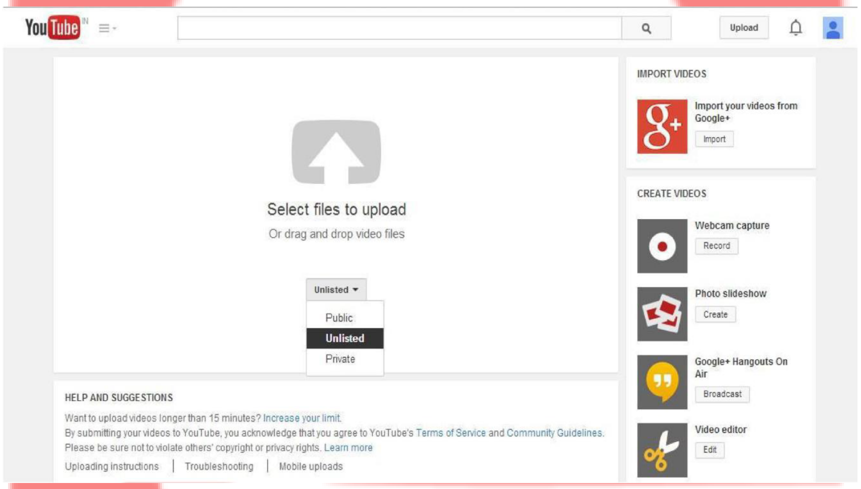
     Remember this json file provides the co-ordinates at which the your drone holds its position. Please note, your drone should go to the exact **x,y & z** co-ordinate that is present inside the JSON file.
  5. If you have followed the method suggested in Task 3.3, submit the following items as well.
     (a) The RoI (Region of Interest) detection script. Rename it to ***roi_detector.py*** if not the same already.
     (b) Saved RoIs. As mentioned in Task 3.3. You can store the generated RoIs in any form. May that be in pickle or json, etc. Rename it to ***rect_info.<ext>***, where <ext> is the format used for storing RoIs.
  6. If you have not used the method mentioned above, and/or have additional python files that you would like to submit, please include them in the submission as well.

6.3 Video:

- Upon verifying that your task is complete, record a video using a screen recorder like simple-screen recorder or kazam as mentioned in section 4, Expected Output.
- The video should be one continuous take and should not be edited in any manner. Teams uploading an edited video will be disqualified from the competition. **e-Yantra reserves the right to disqualify any team if any foul play is suspected.**

6.3.1 Uploading video/s on YouTube:

- Upload a one-shot continuous video with the title eYRC#SR#Task5_OrgConfig#<TeamID> (For example: If your team ID is 123 then, save it as eYRC#SR#Task5_OrgConfig#123)
- Please note that while uploading the video on YouTube select the privacy setting option as Unlisted as shown in Figure 3. You need to upload the video as instructed on the portal.



Want to upload videos longer than 15 minutes? Increase your limit.
By submitting your videos to YouTube, you acknowledge that you agree to YouTube's Terms of Service and Community Guidelines.
Please be sure not to violate others' copyright or privacy rights. Learn more

Figure 3: Uploading unlisted video on YouTube

6.4 Overview:

1. Please place these files inside a .zip file before uploading:

   - bag file
   - Python code and dependencies, refer **Python Scripts and dependent files:** to know more.
   - Miscellaneous files (if any)

   Your final .zip output must be of the following structure:
   <team_id>_5.zip

   - ***SR_<team_id>_5.bag***
   - **scripts/**
     - ***position_hold.py***
     - ***cell_coords.json***
     - ***beacon_detector.py***
     - ***scheduler.py***
     - ***roi_detector.py*** Optional
     - ***rect_info.<ext>,*** where <ext> is the format of your choosing to save the RoIs. Optional.
     - Any other script which you have use in your task.

2. You must upload the video to YouTube as mentioned in the above instructions and instruction available on portal. You will have to submit the video link on the portal.

Please follow the naming convention strictly as specified in each step. Failure to do so may lead to repercussions.

Instructions for uploading the folder will be provided on portal.

Good Luck!!!