I_Hack Software Track Problem Statements

H1: Reimagine digital world experience

Reimagine digital world experiences that brings together processes, user experiences and technologies in a novel and innovative manner that significantly improves productivity, engagement and user satisfaction. For e.g.,.,

- Reimagine a live sports viewing experience in a digital world leveraging 5G network characteristics and cognitive intelligence to create a WoW factor to the in-stadium viewers and online viewers.
- Application redesign, video delivery service architecture redesign, Application to Network interface redesign etc. could be potential areas to explore.
- In your design, consider cost factors for optimizing your algorithm – for example, 4K video is more expensive than 2K video. Consider how your approach would minimize cost without impacting user experience.
- You can assume the availability of AR/VR headsets, HD TV screen, Video files/streams in 2K (HD) and 4K video format. Assume availability of 10 video streams in different formats. While applying for the contest, you may ask for anything else that you may need, but there is no guarantee that those would be made available.
- You are free to choose any open source software libraries and utilities to create a revamped user experience.

H2: Computer Vision Challenge

Each team has to come up with a solution in 30 hours.

1. Map generation from multiple camera streams – generate a single aggregated map of a scene that is being observed using 2 or more non-stationary mobile phone cameras. Consider indoor scenes such as corridors and rooms in at least 3 adjacent floors of a hostel or outdoor scenes comprising at least 4 departments in IIT Bombay campus.

1. Given a set of depth and color images of one or more objects (front, top, side view, etc.), find if the objects are present in a test image set (which has both color and depth images of multiple occluded objects) – images will be provided.

You can use any available open source software or utilities.

H3: Mobile Manipulators

Mobile Manipulator pick problems

Assume we are developing solutions using mobile manipulators to pick and transport items for retail, warehouse and manufacturing shop floor applications. Initially the mobile manipulator maps the area under operation to create the map. After which one can set goals on this map to make the manipulator to go to a location to do required operation. In these scenarios few of the generic problems to be solved are listed as below.

1. **Problem Statement:** Development of visual tool to identify objects in a Robot SLAM map. We should have an option to select the type of object (for this hackathon we can assume standard objects like square or rectangle). In the given map white represents empty space and black represents obstacles/filled space. Any open source tools can be used.

Input:

1. SLAM map in pgm format
2. Number of objects to be identified in the map
3. Types of objects to be identified (example racks). We should have an option in visual tool to select the type of object.

Output format:

1. Coordinates of object in (x, y, orientation) format.
2. Output saved in a file with comma separation (object type, object ID, along with x, y, orientation of relevant points on the object)
3. Self-adjustment of click point (optional i.e. turn off available) after the click to nearest corner of black area.

2. Problem Statement: We need to provide the pose information as a goal to the robot when it is supposed to pick an item. Generation of robot pose(s) to pick items from racks. For example the warehouse typically has racks with multiple shelves.

Input:

1. SLAM map.

2. Number of rack and Shelf with their positions identified in the map in the form of comma separated file (object type, object ID along with x, y, orientation of relevant points on the object)
3. Arm reach
4. Min distance of robot from shelf that needs to be maintained while picking up items,
5. Planogram information of items on the shelves (example which item is kept where in the shelf)

Output:

1. Pose to pick each item in the rack/ shelf populated against the object ID of each object type (it is possible that same pose could be used to pick multiple objects. For simplicity let us assume that there are n stops (with poses) required to pick all objects in the rack. You need to determine these n poses and map each of the items in the rack against one of these poses)
2. Simulation to demonstrate the correctness of the above

Example image of a rack is as below: