Data Collection and Annotation

Ego-Perspective Setup

In the case of ego perspective setup, we are using a single D435i, we could collect.

Setup (Ego-Perspective)

- We are using `Ubuntu 20.04` and `ROS Noetic` to record data.
- We have 1 D435i camera and we want to collcet the following topics:
  - Compressed Color Images and camera intrinsics:
    - `.../color/camera_info`
    - `.../color/image_raw/compressed`
  - Raw Aligned depth to color:
    - `.../aligned_depth_to_color/camera_info /camera/`
    - `.../aligned_depth_to_color/image_raw`
  - IMU information (gyroscope and accelerometer)
    - `.../accel/imu_info`
    - `.../camera/accel/sample`
    - `.../camera/gyro/imu_info`
    - `.../camera/gyro/sample`

Running the ROS Publisher (Ego-Perspective)

- First, we need to launch the `realsense2_camera` with the launch setup `rs_camera.launch` this will configure find a connected camera to the PC and launch it with the configured parameters in `rs_camera.launch`.
- We could override the default argument values in this launch file by passing them as part of the command

```
roslaunch realsense2_camera rs_camera.launch align_depth:=true
depth_width:=640 depth_height:=480 depth_fps:=30 color_width:=640
color_height:=480 color_fps:=30 enable_color:=true enable_infra:=true
enable_infra1:=true enable_pointcloud:=false enable_gyro:=true
enable_accel:=true
```

Recording data (Ego-Perspective)

To record the data we use rosbag

```
rosbag record /camera/accel/imu_info /camera/accel/sample
/camera/align_to_color/parameter_descriptions
/camera/align_to_color/parameter_updates
/camera/aligned_depth_to_color/camera_info
/camera/aligned_depth_to_color/image_raw /camera/color/camera_info
```

```
/camera/color/image_raw/compressed /camera/extrinsics/depth_to_color
/camera/extrinsics/depth_to_infra1 /camera/gyro/imu_info
/camera/gyro/sample /camera/motion_module/parameter_descriptions
/camera/motion_module/parameter_updates /tf /tf_static -o ego_recording.bag
```

Topic monitoring and visualization

- We could use `rostopic hz topic1,topic2,...` to monitor the frequency of the messages over time.
- We could use `rviz` to visualize data by adding the image topics as Image. We could also set `enable_pointcloud:=true` then visualize the 3D pointcloud data with respect to the `camera_link` frame.

Tabletop Setup

In the case of the tabletop setup we are using 6 D435 cameras and 2 D435i. However, the IMU sensors here are not useful sense the cameras are stationary.

Setup (Tabletop)

- We are using two `OptiPlex 7040 micro` PCs each with `Ubuntu 20.04` to support 8 cameras with compressed color and uncompressed aligned depth. However, in case we need 4 or less views we could use 1 PC only.
- We have 1 D435i camera and we want to collcet the following topics:
    - Compressed Color Images and camera intrinsics:
        - `.../color/camera_info`
        - `.../color/image_raw/compressed`
    - Raw Aligned depth to color:
        - `.../aligned_depth_to_color/camera_info /camera/`
        - `.../aligned_depth_to_color/image_raw`

Running the ROS Publisher (Tabletop)

We need to launch the `realsense2_camera` with the launch setup `rs_multiple_devices.launch` and pass the pass the camera serials we want to publish from. parameters in `rs_multiple_devices.launch`. For convience we modify the `rs_multiple_devices.launch` and create a copy named `rs_multiple_devices_setup_all.launch` which contains the serials we have written as default values. The command would be as follows:

```
 roslaunch rs_multiple_devices_setup_all.launch align_depth:=true
depth_width:=640 depth_height:=480 depth_fps:=30 color_width:=640
color_height:=480 color_fps:=30 enable_color:=true enable_infra:=false
enable_infra1:=false enable_pointcloud:=false enable_accel:=false
enable_gyro:=false filters:=spatial,temporal
```

Recording Data (Tabletop)

Similar to the Ego-Perspective setup we are using rosbag, however here we will have 8 times the number of topics. Therefore, we use `rostopic list` to list all the available topics, and `grep` with a regular expression to filter the topics we want. So for data recording the command becomes as follows:

```
rosbag record $(rostopic list | grep
"aligned_depth_to_color/image_raw$\|/color/image_raw$\|/color/camera_info$\
|/aligned_depth_to_color/camera_info$") -o tabletop_recording.bag --split -
-size 3500 -b 0
```

Note that we setup `-b 0` to set the bag buffer to unlimited, and `--split --size 3500` to split the bags in chunks of 3500MB

Topic monitoring and visualization (Tabletop)

We could use `rostopic hz` to monitor the frequency of the messages:

```
rostopic hz $(rostopic list | grep
"aligned_depth_to_color/image_raw$\|/color/image_raw$\|/color/camera_info$\
|/aligned_depth_to_color/camera_info$")
```

Also we can use `rviz` to visualize the data in a similar way to the Ego-Perspective setup.

Data Annotation

CVAT

An open source annotation tool which allows semi automatic image annotation.

MiVOS

Mivos is an interactive video segmentation tool which allows fast video segmentation.

Ontologies

This topic was not covered during the session. We could represent our knowledge about the system we have using an Ontology which defines properties and relationships for/between objects. You could use Protege to create your own knowledge base and access it in python using libraries like `owlready2`.