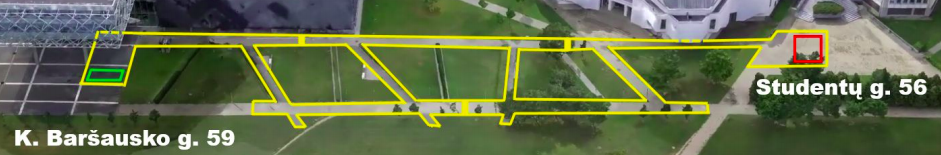
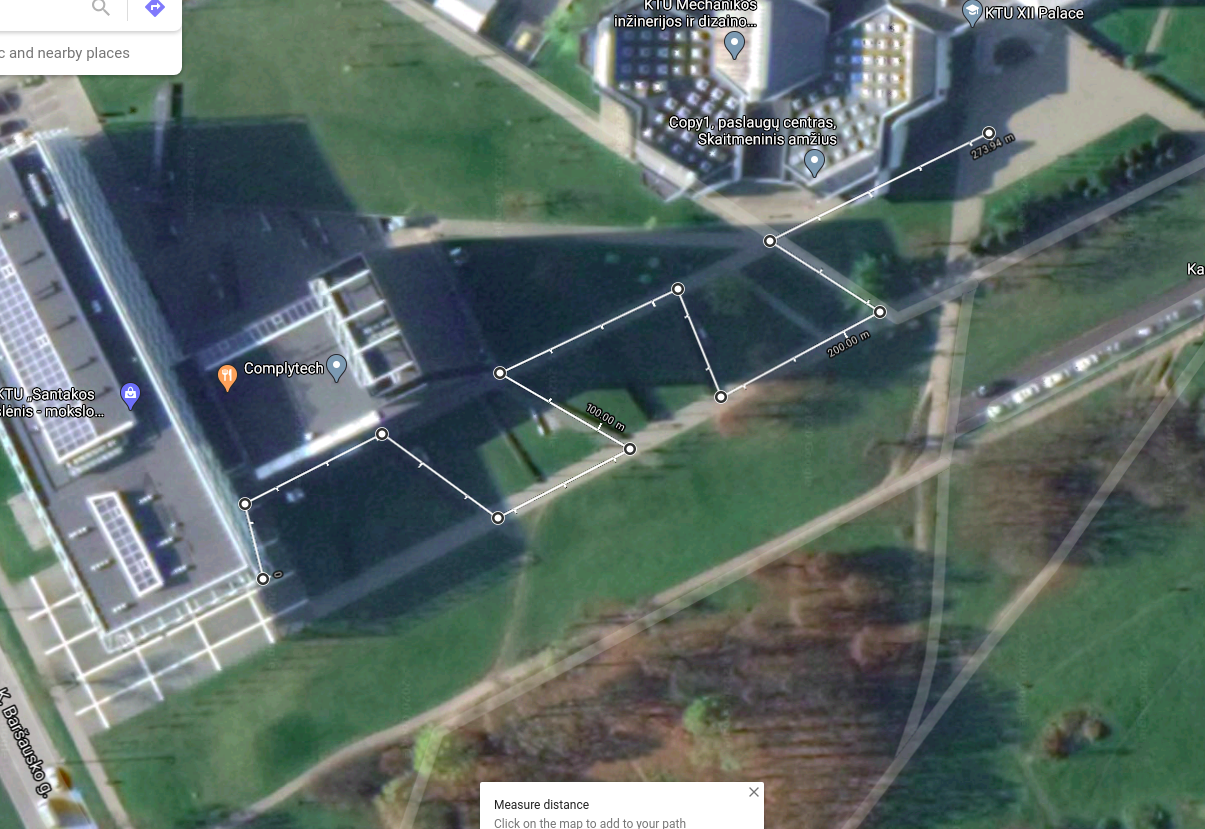
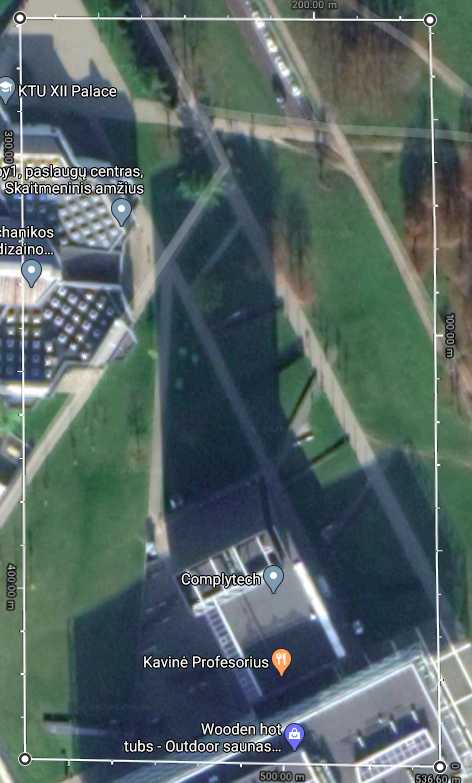
Robot

# Event





One way distance: 273m

Total distance: **546m**

Single run time: **1h**

Time for maintenance: **30min**

Corridor narrows to **125cm** near the ramp.

You can set GPS perimeter **1h - 15m** before start

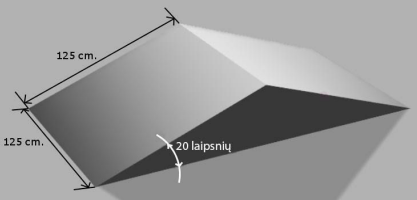
Velocity:

Pure minimum avg\_vel = 0.152m/s

15min to spare: avg\_vel = 0.202m/s (Minimal)

30min to spare: avg\_vel = **0.303m/s**  (Good)

45min to spare: avg\_vel = 0.627m/s (Speed)

Search area:

~10x10m

Cosmap\_2d:

Not rotated:

180x100m 18000m2

Cell size 0.025m: 7200x4000 -> 110MiB (real 28.8MiB)

Cell size 0.050m: 3600x2000 -> 27.5MiB (real 7.0MiB)

Rotated:

170x30m 5100m2

Cell size 0.025m: 6800x1200 -> 31.1MiB

Cell size 0.050m: 3400x600 -> 7.8MiB

**Actually used map** (padded about 7m in all sides to account for uncertainty):

124x175m 21700m2

**Cell size 0.1m: 1240x1750 -> 2.2MiB**

# New PC setup

(not complete; cd directories are example and might be different; not all steps are necessary)

* **Install Arduino IDE**
  + sudo snap install arduino # For newest version (on apt it’s old 2:xx)
  + <https://ubuntu.com/tutorials/install-the-arduino-ide#2-installing-via-a-tarball>
  + Setup STM32 stuff
    - <https://maker.pro/arduino/tutorial/how-to-program-the-stm32-blue-pill-with-arduino-ide>
* **Install vscode**
  + From Ubuntu software
  + Extensions: Arduino, C/C++, Git Graph, Python, Todo Tree
  + ROS setup: <https://github.com/RoboGnome/VS_Code_ROS#1-vs-code-extensions>
* **Install PyCharm Community**
  + From GUI Ubuntu Software
  + Increase max max inode wathces <https://unix.stackexchange.com/questions/13751/kernel-inotify-watch-limit-reached>
* **Reduce Ubuntu padding** by creating file at ~/.config/gtk-3.0/gt:

|  |
| --- |
| /\*  Reducing padding \*/  .default-decoration {  min-height: 0px;  padding: 0px; }  .default-decoration .titlebutton {  min-height: 0px;  padding: 0px; }  .header-bar.default-decoration {  padding-top: 0px;  padding-bottom: 0px;  }  .header-bar.default-decoration .button.titlebutton {  padding-top: 0px;  padding-bottom: 0px; }  /\* No line below the title bar \*/ .ssd .titlebar {  border-width: 0px;  box-shadow: none; } |

* **Set title to terminal tab** (edit ~/.bashrc)**:**

|  |
| --- |
| # Renaming terminal tab function title() {  if [[ -z "$ORIG" ]]; then  ORIG=$PS1  fi  TITLE="\[\e]2;$\*\a\]"  PS1="${ORIG}${TITLE}" } |

* **Install ROS Melodic**
  + <http://wiki.ros.org/melodic/Installation/Ubuntu>
  + Update project dependencies

|  |
| --- |
| # Check missing dependencies rosdep check --from-paths ~/Documents/SamanaAutonomousRobot/ROS/samana\_ws/src/ --ignore-src # Check all dependencies (only installed?) rospack depends samana # Simulate what dependencies will be installed rosdep install --simulate --from-paths ~/Documents/SamanaAutonomousRobot/ROS/samana\_ws/src/ --ignore-src # Install missing dependencies rosdep install --from-paths ~/Documents/SamanaAutonomousRobot/ROS/samana\_ws/src/ --ignore-src # Or simplest form rosdep install samana --ignore-src |

* + Manual dependencies

|  |
| --- |
| sudo apt install nvidia-container-toolkit sudo apt install ros-melodic-sbpl-lattice-planner  sudo apt install feh |

* **Give usb permissions when** connected all Arduinos

|  |
| --- |
| sudo chmod 0777 /dev/ttyUSB\* |

* **Install Roboware Studio**
  + Files here: <https://github.com/TonyRobotics/RoboWare>
  + On ERROR: bash: catkin\_make: command not found
    - Click this icon
    - ctrl + f and find “catkin\_make” in tasks.json
    - Add source ros to before catkin\_make. Also add to release task
    - It should look like:

|  |
| --- |
| "taskName": "Debug", "args": [  **"source /opt/ros/melodic/setup.bash;** catkin\_make -C ${workspaceRoot} -DCMAKE\_BUILD\_TYPE=Debug" ], |

* **Make symbolic links of Arduino libraries** to actual path where Arduino libraries are read (might need editing make\_links to point to right location)
  + cd /home/combinacijus/Documents/SamanaAutonomousRobot/Arduino/
  + sh make\_links.sh
* **Copy udev rules** from Other Files

|  |
| --- |
| cd /home/combinacijus/Documents/SamanaAutonomousRobot/OtherFiles/UbuntuCofigs/UdevRules sudo cp 49-stlinkv2\* 99-usb-serial.rules /etc/udev/rules.d/ sudo udevadm control --reload |

* **Install Docker (and** [**run here**](#_gu9l0cdq2ll5)**)**
  + <https://docs.docker.com/engine/install/ubuntu/>

|  |
| --- |
| # Add permissions to edit Docker config file sudo chown "$USER":"$USER" /home/"$USER"/.docker -R sudo chmod g+rwx "/home/$USER/.docker" -R  # Add the package repositories # On Error when running docker run below commands (should be found online): # docker: Error response from daemon: could not select device driver "" with # capabilities: [[gpu]]. distribution=$(. /etc/os-release;echo $ID$VERSION\_ID) curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add - curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list sudo tee /etc/apt/sources.list.d/nvidia-docker.list  sudo apt-get update && sudo apt-get install -y nvidia-container-toolkit sudo systemctl restart docker  # Nvidia driver dependency sudo apt install nvidia-driver-440  sudo apt install nvidia-container-toolkit  # Download image for CUDA docker pull combinacijus/imageai:compute3.0 |

* **Git and git LFS**
  + <https://git-lfs.github.com/>

|  |
| --- |
| # Git sudo apt-get install git  # git LFS  curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash sudo apt install git-lfs |

* **Build catkin workspace setup**
  + sudo apt-get install make
  + On Error: Makefile:140: recipe for target 'all' failed

On Error failed catkin\_make: make[2]: \*\*\* No rule to make target '**/usr/lib/x86\_64-linux-gnu/libGL.so**', needed by '/home/combinacijus/Documents/test\_ws/devel/lib/librviz\_imu\_plugin.so'. Stop.

<https://github.com/RobotLocomotion/drake/issues/2087#issue-148166827>

|  |
| --- |
| # Check does this link point to existing file or that file to existing file etc readlink /usr/lib/x86\_64-linux-gnu/libGL.so # Remove this link if not sudo rm /usr/lib/x86\_64-linux-gnu/libGL.so # Find where libGL.so.1 is located sudo find / -name "libGL.so\*" # Relink it (might need changing directories) sudo ln -s /usr/lib/x86\_64-linux-gnu/libGL.so.1 /usr/lib/x86\_64-linux-gnu/libGL.so |

* **git clone [Samana repository] and build**

|  |
| --- |
| mkdir /home/combinacijus/Documents/SamanaAutonomousRobot  cd /home/combinacijus/Documents/SamanaAutonomousRobot git clone https://github.com/Combinacijus/Samana-Autonomous-Robot.git cd ./ROS/samana\_ws catkin\_make |

* **~/.bashrc**

|  |
| --- |
| # ROS source "/opt/ros/melodic/setup.bash" source "/home/combinacijus/Documents/SamanaAutonomousRobot/ROS/samana\_ws/devel/setup.bash" source "/home/combinacijus/Documents/ROS/learning\_tf\_ws/devel/setup.bash" export TURTLEBOT3\_MODEL=waffle\_pi  stty -ixon export PATH=/usr/local/cuda-10.0/bin${PATH:+:${PATH}} export LD\_LIBRARY\_PATH=/usr/local/cuda-10.0/lib64${LD\_LIBRARY\_PATH:+:${LD\_LIBRARY\_PATH}} |

* **Free up space on /home partition by symlinking** to / partition cache and other directories

|  |
| --- |
| # Directory setup sudo mkdir /usr/home-partition sudo chown combinacijus /usr/home-partition/ sudo chmod 775 /usr/home-partition/ cd /usr/home-partition  # Free up space sudo baobab # To find disk usage # Create directory for VSCode workspace files sudo mkdir /usr/home-partition/.config/Code/User/workspaceStorage/  # Via gui copy files to /usr/home-partition/<filepath-after-home>  # Symlinks. .cache is regenerating so run fast (after redeleting) sudo ln -s /usr/home-partition/.config/Code/User/workspaceStorage/ ~/.config/Code/User/  sudo ln -s /usr/home-partition/.cache/ /home/combinacijus/ |

# Git

* To push single branch to remote: (origin = Samana-...

|  |
| --- |
| git checkout feature\_x git push origin -u feature\_xg |

* + use -f if you need to overwrite backup branch
* Undo last commit:

|  |
| --- |
| git reset --soft HEAD~1 |

* To move commits from backup\_branch to other branch:
  + Go to new branch
  + Cherry pick commits you want
  + Delete backup\_branch
  + Create backup\_branch

**Git log (A DOG):**

|  |
| --- |
| git log --all --decorate --oneline --graph # OR git config --global alias.adog "log --all --decorate --oneline --graph" git adog |

**Merge many commits:**

|  |
| --- |
| git reset HEAD@{n} # n found from git reflog git commit |

**delete branch local and remote:**

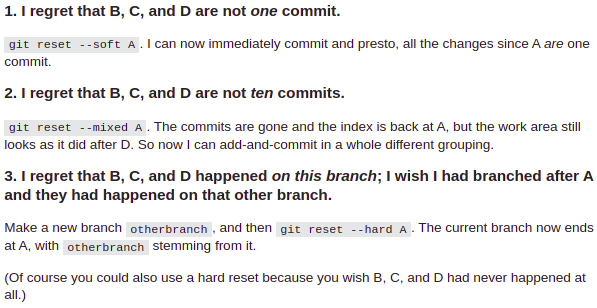
|  |
| --- |
| # delete branch locally git branch -d localBranchName  # delete branch remotely git push origin --delete remoteBranchName  git push origin :remoteBranchName |

**git lfs:**

|  |
| --- |
| git lfs track "\*\*/\*.lfs" # This generates .gitattributes Or edit it git add .gitattributes git add file\_to\_be\_in\_lfs git lfs ls-files # Should list all LFS file already pushed and newly staged git lfs status # Should show files with LFS: tag |

### git reset

Good for tidying up commits to single commit or to many commits

A - B - C - D <- HEAD

**git reset vs git rm --cached**

**git rm --cached,** puts specified files into untracked, and stages their removal for the next commit.

**git reset --mixed**, simply moves the file into 'unstaged', but keeps its old version in repo (opposite of git add).

# GPS

* Find IP in terminal write ***ip route*** and find last ip *scope link src* ***192.168.0.102*** *metric* append **:11311/**
* Connect via ROS Sensors Driver on Android with same IP eg. http://192.168.0.102:11311
* No error should be shown

|  |
| --- |
| # In ROS run these commands  export ROS\_MASTER\_URI=[http://192.168.0.102:11311/](http://192.168.0.100:11311/) # Not necessary? roscore rosrun topic\_tools trsform /android/fix /android/fix/fixed sensor\_msgs/NavSatFix 'sensor\_msgs.msg.NavSatFix(header=std\_msgs.msg.Header(seq=m.header.seq,stamp=m.header.stamp,frame\_id="gps"), status=m.status, latitude=m.latitude, longitude=m.longitude, altitude=m.altitude, position\_covariance=m.position\_covariance, position\_covariance\_type=m.position\_covariance\_type)' --import sensor\_msgs std\_msgs |

* Lowest covariance on gps module at night: 15500.25 (very bad)
* Lowest covariance on a smartphone at night: 18.38 (fair?)

# General programming

* IMPORTANT: sync clocks with ROS clock
* F() Macro on Arduino printing. To save SRAM
* nh.getHardware()->setBaud(1000000);
* rosrun rosserial\_arduino serial\_node.py \_baud:=115200 \_port:=/dev/ttyUSB0
* delay(1);
* Be careful with rosserial because it can desync if data is sent too fast
* 92% memory usage causes instability
* Lost sync with device, restarting…
  + Lower baud rate
* To build ROS library for Arduino

1. build samana\_msgs

|  |
| --- |
| source devel/setup.bash rm -r ~/Documents/SamanaAutonomousRobot/Arduino/libraries/ros\_lib rosrun rosserial\_arduino make\_libraries.py ~/Documents/SamanaAutonomousRobot/Arduino/libraries/ |

1. If can’t include headers first include ros.h
2. [wiki.ros.org/rosserial\_arduino/Tutorials/Arduino%20IDE%20Setup](http://wiki.ros.org/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup)

* Further if adding customized ros\_lib for a node:
  + In Samana project main folder */Arduino* modify ***make\_links.sh*** to create a symbolic link of *ros\_lib\_newnode* to a Arduino/libraries folder on computer (place where IDE reads libraries) e.g.

|  |
| --- |
| ln -sv /home/combinacijus/Documents/SamanaAutonomousRobot/Arduino/libraries/ros\_lib\_odom/ /home/combinacijus/Documents/Arduino/libraries/ |

* If ROS workspace change or adding new

|  |
| --- |
| sudo subl .bashrc source "/home/combinacijus/Documents/SamanaAutonomousRobot/ROS/samana\_ws/setup.sh" |

* + .../build/CMakeCache.txt change all directories to new one
  + Also in RoboWare change settings.json for autocomplete
* If node tried to send before init “Tried to publish before configured”:
  + Build project (catkin\_make)
  + Source devel/setup.sh
  + Check rosmsg list if your package messages are loaded
* To accept arduino node to roscore

|  |
| --- |
| sudo chmod 666 /dev/ttyUSB\* rosrun rosserial\_arduino serial\_node.py \_baud:=10000000 \_port:=/dev/ttyUSB0 \_recal:=false |

* UDEV rule for separating Aruduino depending on devpath (usb port)

|  |
| --- |
| SUBSYSTEM=="tty", ATTRS{idVendor}=="1a86", ATTRS{idProduct}=="7523", SYMLINK+="ttyUSB-arduino%s{/devpath}" |

* + This gives names like: /dev/ttyUSB-arduino2.1 /dev/ttyUSB-arduino2.3.3

# General electronics:

* **IMPORTANT:** Capacitor on input to eliminate power loses

# IMU BNO055

* SCL: A5; SDA: A4, 5V, GND
* Human ride: max angular velocity 300deg/s
* Human ride: max acceleration 3m/s^2
* BNO055 loses calibration when space between hoverboard is <20cm (30cm from any electronic would be nice)
* On system calibration loss. 360 + 180 - 180 is advised to recalibrate IMU
* IMPORTANT: On magnetometer calibration degradation one wheelie to different headings might calibrate it
* **IMPORTANT:** After loosing power IMU won’t recover. Needs extra code. Problem with Wire.h library in twi.cpp file, issue: <https://github.com/panStamp/arduino_avr/issues/1>. Maybe needs extra Arduino to reset via pin.
* PROBLEM: Heading is not reliable
* Baud rate could be lowered to 57600 but it will reduce update speed from 100hz to 73hz

# Distance sensors

* Max distance 4-6m at home
* Need spacer in front so minimum distance wouldn’t be 0. Else it would cause bugs
* SRF04 will only listen to echo if it was triggered
* Ultrasonic sensors on other robots might trigger own sensors therefore giving shorter distance to the obstacle
* IMPORTANT: deal with bump sensors when angled
* avrdude: stk500\_recv(): programmer is not responding. But will upload via Arduino IDE
* IMPORTANT: ROS: check each sensor max distance. If it’s too big map won’t be accurate it will have wall at the end of sonar cone

# Hoverboard

* IMPORTANT: There is inactivity timeout on hoverboard firmware. It probably should be disabled for competition
* Main battery voltage screen shows **0.4V lower voltage**
* Battery:
* 3.6V Low voltage no beep
  + 3.5V Almost empty battery; beeps
  + 3.37V Turns offs when almost no speed
  + Charging speed. no\_load=35.1V to no\_load=38.6V in 1hr; to 41.6V 1h45m

# Arm

* IMPORTANT: if motors disables on their own it might be because overcurrent protection
* If motors don’t work all manual switches might be on

Arm data:

|  |  |  |
| --- | --- | --- |
|  | Dry thread | With bag (worst grab) |
| Lifter up | 48s | 58s |
| Lifter down | 54s | -;just a li |
| Grabber open | 12s | - |
| Grabber close | 12s | - |

# Object recognition

* Pretrained YoloV3: <https://github.com/OlafenwaMoses/ImageAI/releases/tag/essential-v4>
* *You can keep the images at their original size. The training network will automatically re-scale the image to sizes between* ***288 to 488*** *as it runs through many experiments.*
* Too large images will exhaust GPU memory
* On CPU 0.34 - 0.42 FPS on 100%; on GPU
* GPU worked on nvidia-smi Driver Version: ~~390.132~~ 440
* On ERROR: socket.error: [Errno 98] Address already in use. Run lsof -i :5005 and check listening processes and sudo kill -9 <process\_id>
* To connect **Android phone as IP** camera create hotspot on computer and connect to it via phone's wifi
  + To create **hotspot** go to: Wifi Settings -> Menu button at the top right corner -> Turn on Wi-Fi Hotspot

### Google Colab local runtime

|  |
| --- |
| jupyter notebook --NotebookApp.allow\_origin='https://colab.research.google.com' --port=8888 --NotebookApp.port\_retries=0 --allow-root |

### Docker

|  |
| --- |
| sudo docker run -it --rm --gpus all --net host -v /home/combinacijus/Documents/SamanaAutonomousRobot/Python/GoldBagDetector/:/notebooks/GoldBagDetector -v /home/combinacijus/Desktop/ros-logs/:/home/combinacijus/Desktop/ros-logs/ -w /notebooks/GoldBagDetector --name imageai combinacijus/imageai:compute3.0  sudo docker run -it --rm --gpus all --net host --device /dev/video2 -v /home/combinacijus/Documents/SamanaAutonomousRobot/Python/GoldBagDetector/:/notebooks/GoldBagDetector -w /notebooks/GoldBagDetector --name imageai combinacijus/imageai:compute3.0  cd /notebooks/GoldBagDetector/  python GoldBagDetector.py  # Run ROS script (TCP/IP server side)  # Set laptop to performance mode  feh -R 0.7 /home/combinacijus/Documents/SamanaAutonomousRobot/Python/GoldBagDetector/image.jpg # For image debugging  sudo docker run -it --gpus all -v ~/Documents/SamanaAutonomousRobot/Python/GoldBagDetector/:/notebooks/GoldBagDetector imageai  sudo docker run --gpus all -it --rm tensorflow/tensorflow:1.12.0-gpu /bin/bash  sudo docker run --gpus all -it -v /host\_path:/container\_path tensorflow/tensorflow:1.12.0-gpu /bin/bash   sudo docker images  sudo docker ps -a # Show running containers  sudo docker start -ai CONTAINER  sudo docker commit CONTAINER NEW\_IMAGE\_NAME  # eg. sudo docker commit focused\_taussig combinacijus/imageai:compute3.0  sudo docker rmi IMAGE  sudo docker stop CONTAINER   sudo docker login # Works  sudo docker login docker.io  sudo docker tag ID combinacijus/imageai  sudo docker push combinacijus/imageai |

### To update Docker image

|  |
| --- |
| sudo docker login sudo docker run -it --name imageai combinacijus/imageai:compute3.0 # In Dokcer run commands to install packages needed # Press Ctrl+D to exit container sudo docker ps -a # Check name or ID # (template but check names before running) sudo docker commit CONTAINER NEW\_IMAGE\_NAME sudo docker commit imageai combinacijus/imageai:compute3.0 sudo docker push combinacijus/imageai:compute3.0 |

ubuntu

* On Error: docker: Error response from daemon: OCI runtime create failed: container\_linux.go:349: starting container process caused "process\_linux.go:449: container init caused \"process\_linux.go:432: running prestart hook 0 caused \\\"error running hook: exit status 1, stdout: , stderr: nvidia-container-cli: initialization error: driver error: failed to process request\\\\n\\\"\"": unknown.

|  |
| --- |
| # Install Nvidia driver and reboot computer sudo apt install nvidia-440 |

### Training new model

1. Take photos with the ‘Open Camera’ app. Around 200 for an object.
2. Set resolution to 640x480.
3. Photos can be taken with bluetooth headphones volume buttons.
4. Folder structure:
   1. dataset1
      1. train
         1. images
            1. IMG001.jpg
            2. IMG002.jpg
         2. annotations
            1. IMG001.xml
            2. IMG002.xml
      2. validation
         1. images
            1. IMG003.jpg
         2. annotations
            1. IMG003.xml
5. Separate images to validation set now or do it automatically randomly later.
6. Rename all images to IMGXTxxx.jpg eg. IMG2T001,jpg IMG2V023.jpg. X - dataset number, T - type, either T for train or V for validation, x - images number or XTxxx - any images name could be default name.
7. Annotate images with **labelImg**. Class name **‘goldbag’**. View-> **Single Class Mode**. View -> **Auto Save Mode**. <https://medium.com/deepquestai/object-detection-training-preparing-your-custom-dataset-6248679f0d1d>
8. Edit ‘‘seperate-validation-images.py’ by reading ‘NOTE:’ comments. Change directories to right ones.
9. Run ‘seperate-validation-images.py’ script to separate around 20% of images and annotations to validation set.
10. Upload data set to Google Drive in GoldBagDetector directory
11. Go to jupyter notebook: <https://colab.research.google.com/drive/17DeoyW0hmTHndsOTblY4hXfw0GV6wFmA#scrollTo=F_8qYPEJt3eg>
12. Change directories noted by ‘NOTE:’ comments
13. Run javascript command to imitate activity so that max limit would be 12hr instead of 90min

<https://stackoverflow.com/questions/57113226/how-to-prevent-google-colab-from-disconnecting>

function ConnectButton(){

console.log("Connect pushed");

document.querySelector("#top-toolbar > colab-connect-button").shadowRoot.querySelector("#connect").click()

}

setInterval(ConnectButton,60000);

1. Run scripts in order to train the model
2. One epoch takes **12min to train**. train: 391, validation: 97, batch\_size=4, GPU

One epoch takes **15min to train**. train: 467, validation: 126, batch\_size=4, GPU

1. Training is done in around 10-16 epochs (over 2h)

# ROS

Python nodes might get **stuck** running after shutting down launch file and would eat a lot of resources in that case:

|  |
| --- |
| kill -9 <PID> |

To visualise range data (launch file):

* Make URDF model
* Find and Load URDF model to param server:

|  |
| --- |
| <arg name="model" default="$(find urdf\_tutorial)/urdf/01-myfirst.urdf"/> <param name="robot\_description" command="$(find xacro)/xacro.py $(arg model)" /> |

* For loading rviz config:

|  |
| --- |
| m<arg name="rvizconfig" default="$(find urdf\_tutorial)/rviz/urdf.rviz" /> |

* Publish joint states (all rotation of the robot etc):

|  |
| --- |
| <node name="joint\_state\_publisher" pkg="joint\_state\_publisher" type="joint\_state\_publisher" /> |

* Publish all TFs of the robot:

|  |
| --- |
| <node name="robot\_state\_publisher" pkg="robot\_state\_publisher" type="state\_publisher" /> |

* Start rviz:

|  |
| --- |
| <node name="rviz" pkg="rviz" type="rviz" args="-d $(arg rvizconfig)" required="true" /> |

* Full example launch file:

|  |
| --- |
| <launch> <arg name="model" default="$(find urdf\_tutorial)/urdf/01-myfirst.urdf"/>  <arg name="gui" default="true" />  <arg name="rvizconfig" default="$(find urdf\_tutorial)/rviz/urdf.rviz" />   <param name="robot\_description" command="$(find xacro)/xacro.py $(arg model)" />  <param name="use\_gui" value="$(arg gui)"/>   <node name="joint\_state\_publisher" pkg="joint\_state\_publisher" type="joint\_state\_publisher" />  <node name="robot\_state\_publisher" pkg="robot\_state\_publisher" type="state\_publisher" />  <node name="rviz" pkg="rviz" type="rviz" args="-d $(arg rvizconfig)" required="true" /> </launch> |

* To check xacro validity in terminal (cd to urdf file directory):

|  |
| --- |
| check\_urdf <(**xacro** samana.xml) |

* XML might be invalid because comment ending with triple minus --->
* On Error failed catkin\_make: make[2]: \*\*\* No rule to make target '**/usr/lib/x86\_64-linux-gnu/libGL.so**', needed by '/home/combinacijus/Documents/test\_ws/devel/lib/librviz\_imu\_plugin.so'. Stop.

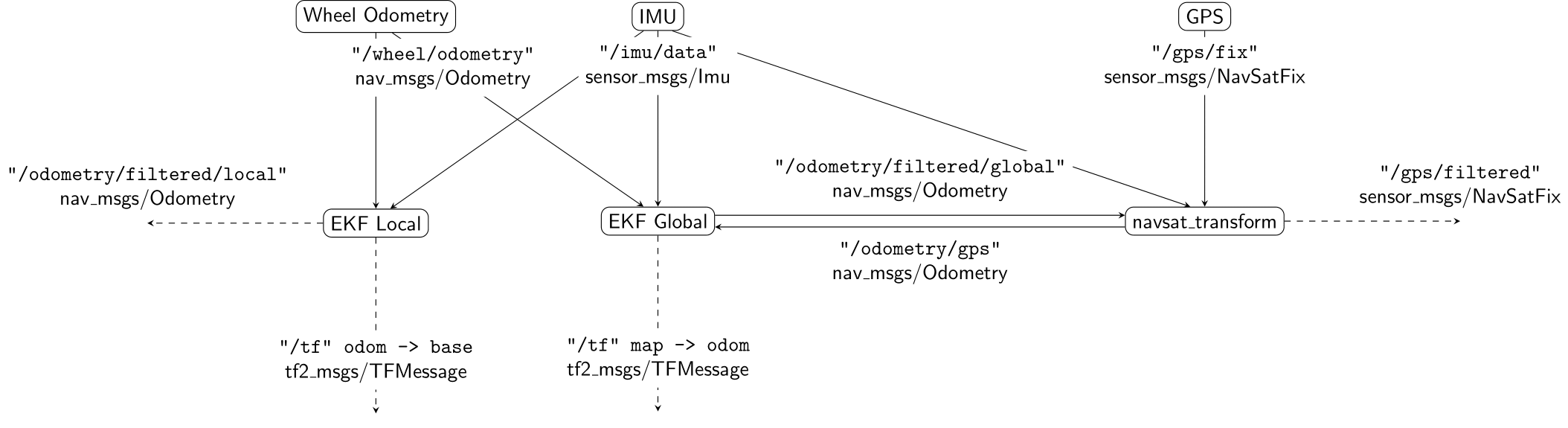
<https://github.com/RobotLocomotion/drake/issues/2087#issue-148166827>

|  |
| --- |
| # Check does this link point to existing file or that file to existing file etc readlink /usr/lib/x86\_64-linux-gnu/libGL.so # Remove this link if not sudo rm /usr/lib/x86\_64-linux-gnu/libGL.so # Find where libGL.so.1 is located sudo find / -name "libGL.so\*" # Relink it (might need changing directories) sudo ln -s /usr/lib/x86\_64-linux-gnu/libGL.so.1 /usr/lib/x86\_64-linux-gnu/libGL.so |

* To filter and extract bag data to the file. bag to file csv

|  |
| --- |
| # cd to directory  rosbag filter bags/Full\_room\_drive.bag bags/full\_room\_sonar\_1.bag "topic == '/bump' or topic == '/sonar' or topic == '/tf' or topic == '/tf\_static'"  rostopic echo -b ./bags/Full\_room\_drive.bag -p /sonar > sonar\_data.csv |

Localization ekf navsat nav\_sat:



# Gazebo:

* Turtlebot3 tutorial: <https://emanual.robotis.com/docs/en/platform/turtlebot3/simulation/#gazebo-3d>

# STM32 in Hoverboard

* To stm32 flash:

SWDIO darker gray wire, GND blue, SWCLK lighter gray

Don’t connect 3.3V!!!

Flashing sequence:

* Test jumper wires on stm32 bluepill board
* Might need to unscrew hoverboard cover
* Connect 3 wires: GND SWDIO SWCLK to board and st-link V2 (NO 3.3V)
* Hold hoverboard power button for all steps forward (or it will turn off)
* Connect st-link to pc. By probing it should show high-density device
* Flash firmware with command below

|  |
| --- |
| watch -n 0.2 st-info --probe # Loops command every 0.2 sec  # cd to project folder  cd ~/Documents/SamanaAutonomousRobot/Other\ Files/hoverboard-firmware-niklas/  # Rebuild code make clean; make  # Flash to stm  st-flash --reset write build/hover.bin 0x8000000 |

* STM32 config file if coreid is unexpected

|  |
| --- |
| /usr/share/opencd/scripts/target/stm32f1x.cfg |

* Ubuntu:
  + Write code in Arduino. Sketch -> Export compiled Binary
  + cd to binary

|  |
| --- |
| *st-flash --reset write <binary\_name>.bin 0x8000000* |

* /opt/STM32CubeProgrammer/bin/STM32CubeProgrammer
* To upload bootloder (On Window easy):
  + Download bootloader [generic\_boot20\_pc13.bin](https://github.com/rogerclarkmelbourne/STM32duino-bootloader/blob/master/binaries/generic_boot20_pc13.bin) (if led pin13):
    - <https://github.com/rogerclarkmelbourne/STM32duino-bootloader/tree/master/binaries>
  + Flash it in any way to the board (did with stlinkv2 and windows st-link util). Worked with both boot pins on 0
  + If DFU device not detected download and install driver (.bat both)
    - <https://codeload.github.com/rogerclarkmelbourne/Arduino_STM32/zip/master>

# Odometry

* Max forward speed 9 rps (rotations per second)
* Max one wheel speed turning 19 rps
* To measure covariances UMBMark test

# Hardware

* **IMPORTANT:** USB hub requires external power supply. Connecting the second laptop USB with double USB A works. Or connect power bank
* Raising the arm with 1kg object uses under 1000mA If pushed could raise around 2kg in the middle of the gripper. To fix it steal plate might be enough
* Weak point of lifter is it’s rack gear because under >1kg of weight it bends a lot
* Lowering without object uses <400mA

# Mapping

* [range\_sensor\_layer](http://wiki.ros.org/action/fullsearch/range_sensor_layer?action=fullsearch&context=180&value=linkto%3A%22range_sensor_layer%22) for making costmap\_2d with sonar sensors
* Some commands

|  |
| --- |
| # To clear costmaps (might not show instantly) rosservice call /move\_base/clear\_costmaps "{}" # To save local costmap rosrun map\_server map\_saver --occ 100 --free 10 -f map\_tmp map:=/move\_base/local\_costmap/costmap |

* symbol lookup error: /opt/ros/melodic/lib//librange\_sensor\_layer.so
  + Probably package is not installed correctly (from source or from apt) or not built or setup.bash not sourced

# Trivia

* Hoverboard automatically turns off after around 3h +-200min of inactivity (no movement command received)
* Size/Dimesions LxWxH 96x68x90cm
* Size/Dimesions 2D WxH 63x58cm
* Hoverboard max speeds:
  + No load (wheel of the ground):
    - **Velocity -4.4m/s - 4.6m/s**
    - **Angular speed -18rad/s - 15rad/s** (-5.7rot/s - 4.8 rot/s)
  + With laptop PID tuned:
    - Velocity 3m/s (tested with velocity pid on, should go a bit faster because pwm was at 80%)
    - Angular speed 7rad/s (only 40% pwm)
* Hoverboard min speeds:
  + With laptop (pid tuned):
    - Velocity 0.02m/s (or maybe even lower with precise input?)
    - Angular speed 0.02rad/s (or maybe lower?)
* Pid brake distance:
  + 0.5-**0.7m/s - 14cm**
  + 0.3m/s - 9cm
* PID acceleration:
  + **6-8 m/s^2**
* Amp draw from the battery at 38V:
  + Minimal: 41 mA | 1.6 W
  + With hoverboard on: 174 mA | 6.6 W
  + With hov and arduinos on: **235 mA | 8.9 W**
  + **Grabber motor** free: 40-120 mA | 1.5-4.7 W (more when mechanism get stuck a bit)
  + **Lifter motor**
    - Free running: 50 mA | 1.9 W
    - Stall: ~500 mA | 19 W
  + **Computer**: 1215 mA | 46.2 W
  + **Total for all electronics** and computation: ~1450 mA | 55.1 W
  + Driving
    - Flat floor walking pace: ~150 mA | 5.7 W
    - High load: >4000 mA | >152.0 W and easily more
* Max speed and rotation: **3.7 m/s** and **7 rad/s**

# Tips

* Use rosrun rosrun rqt\_top rqt\_top for profiling nodes for CPU and RAM usage
* roswtf for general R debugging
* grep -R “text to find” ./path to find recursively text in all files

TODO BEFORE COMPETITION

# TODO

* **BUILD CATKIN\_MAKE WITH RELEASE TAG**
* Fully charge hoverboard
* Calibrate IMU heading
* Check laptop power settings: to be energy efficient, no automatic turn off!
* Hang small bag
* Disable manual arm motors control before autonomous mode
* Check if Arduinos are on right ports
* Check transmitter failsafe behaviour
* Change hard coded temperature on distance sensors Arduino for more accurate distance

|  |
| --- |
| jupyter notebook # Runs Jupyter notebook in a current directory |

# Needed tools and materials

* Screw driver
* Hex tools (for grabber base plates)
* Soldering equipment
* All electronics (voltage regulator, motors, arduinos)
* Wires
* Electrical, masking, double sided tape
* Wrench 8mm?
* Saw for cutting wood
* Plywood and wood material
* Hot glue gun
* Blue foam
* Wood screws
* Scissors, pen
* Two pliers