Robotics for AI

Development lecture 3: Object Recognition

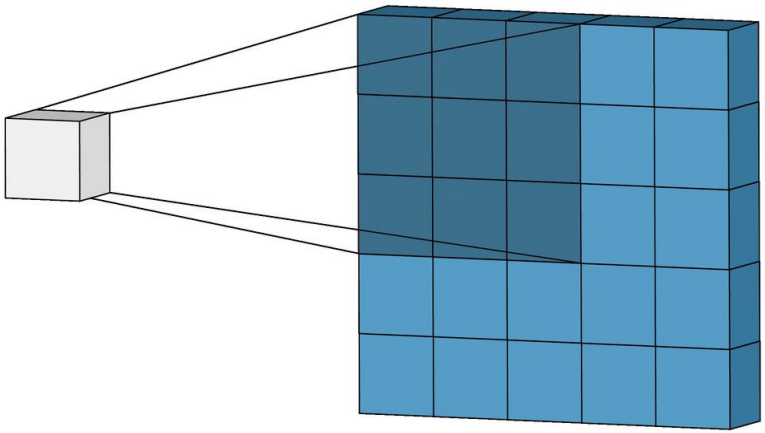
Table of content

Deep Learning

- 2D Convolutions → feature extraction
- Pooling layers → downscaling feature maps (hence less memory needed)
- Multilayer perceptron → classification

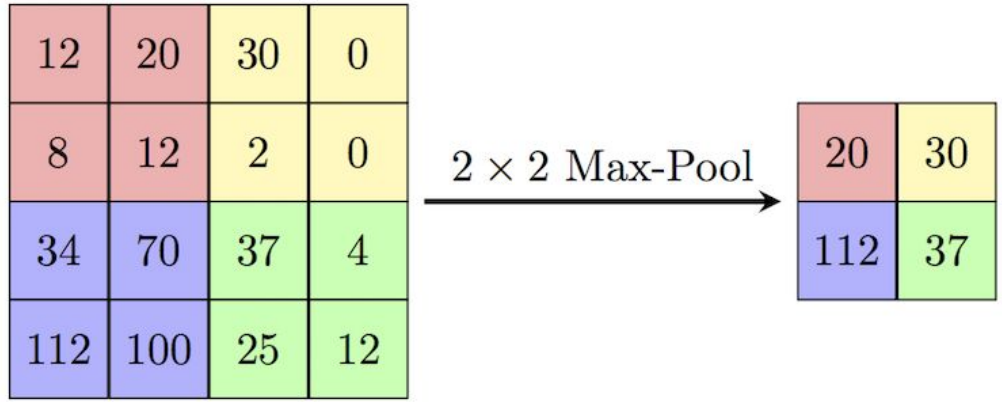
Deep Learning

- 2D Convolutions:
  - Kernel shape
  - Stride
  - Padding
  - Activation functions
    - sigmoid
    - relu
    - Tanh
    - More exist..



Deep Learning

- Pooling layers:
  - Max pooling
  - Average pooling
  - Other option for downsizing:
    - strides > 1 during convolutions



Deep Learning

We have a 30 x 30 x 32 featuremap.

If we want to go to classification / towards an MLP, we have to flatten:

[30 x 30 x 32] featuremap → [30 * 30 * 32] = [28800]

This step can be a huge memory bottleneck!

Deep Learning

So now we have [28800] feature vector.

If the next fully connected layer has 200 units, the required weight matrix will have the shape: [200 x 28800], resulting in 5 760 000 trainable weights!

Make sure your feature maps aren’t too large before flattening.

Deep Learning with Tensorflow and Numpy

So, feature maps and images have a shape of [width, height, channels]

Having multiple images, you will have to use this shape:

[nImages, width, height, channels]

Hence, having 100 images with the size 1024 x 768 and 3 channels (r,g,b):

[100 x 1024 x 768 x 3]

Having 100 output vectors from the CNN, will result in a shape:

[100 x nClasses]

Deep Learning with Tensorflow and Numpy

You can train / run your CNN on batches of data.

Hence:

Input: [batch size, width, height, channels]

Output: [batch size, nClasses]

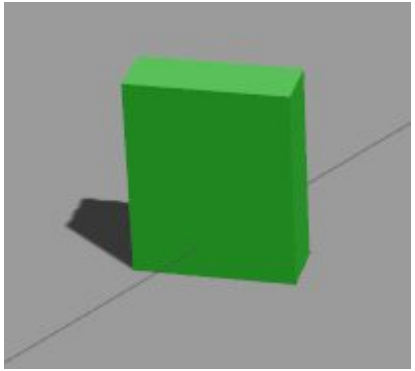
The model will be trained using the average error.

Data gathering:

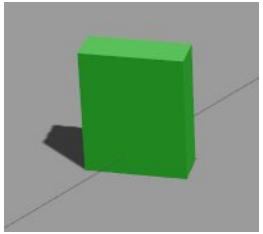
In simulation:

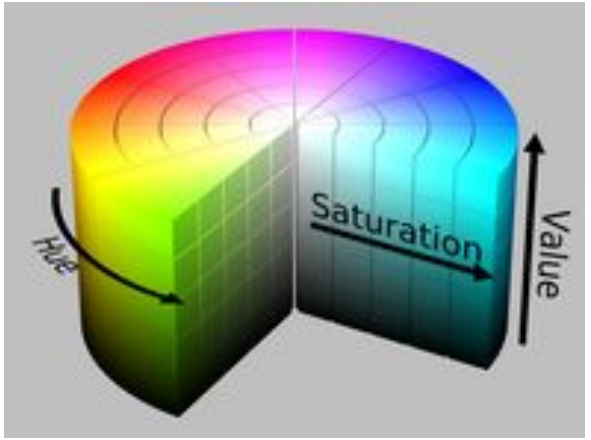
- Spawn object (A script is provided to do this)
- Drive to and approach the objects
- Repeat:
  - Collect the ROI (we provide a tool to do this)
  - Respawn an object

For real objects, it’s already done for you



Data Augmentation





Croppings are not perfect!

For every ROI, make slightly other ROIs from this object.

- Add a bit to a side / remove a bit from a side
- Slightly larger / slightly smaller ROIs

Lighting conditions are never constant in real life!

- Convert the image to HSV (hue, saturation and value) color space
- Use slightly different Value / Saturation values to augment brightness
- Convert back to RGB color space

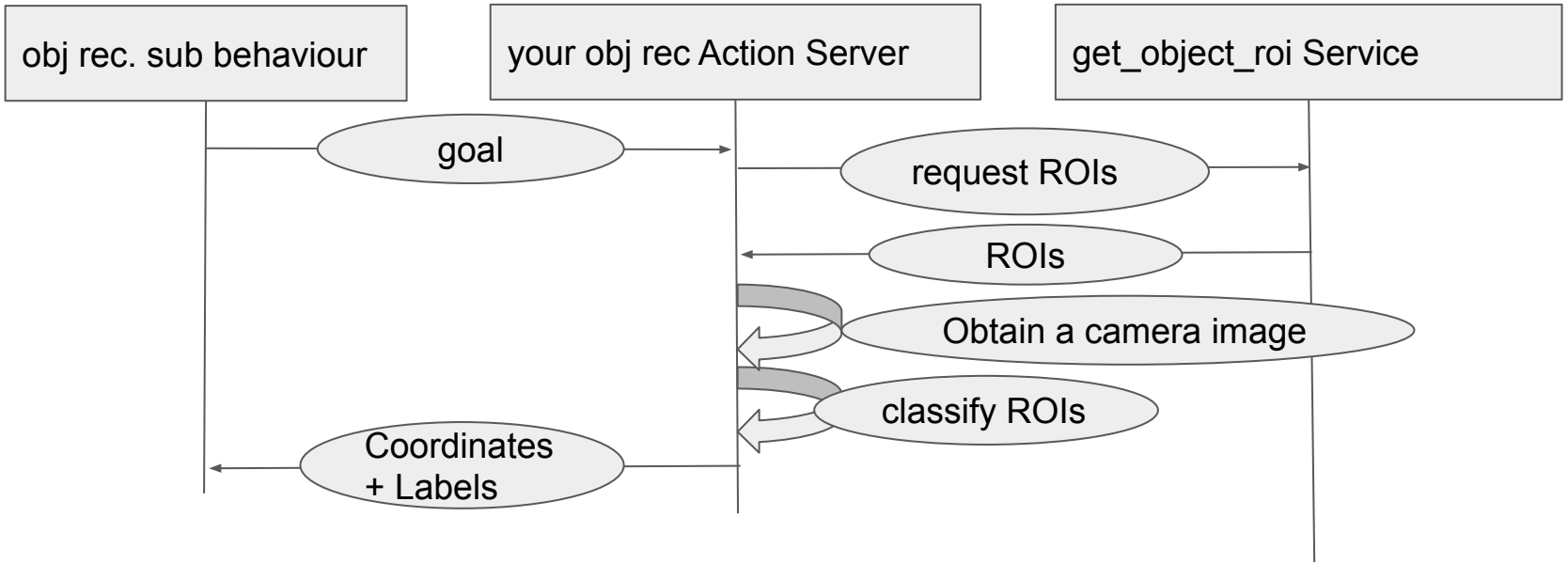
Numpy tutorial

Switching to other file to present…

Deep Learning with Tensorflow:

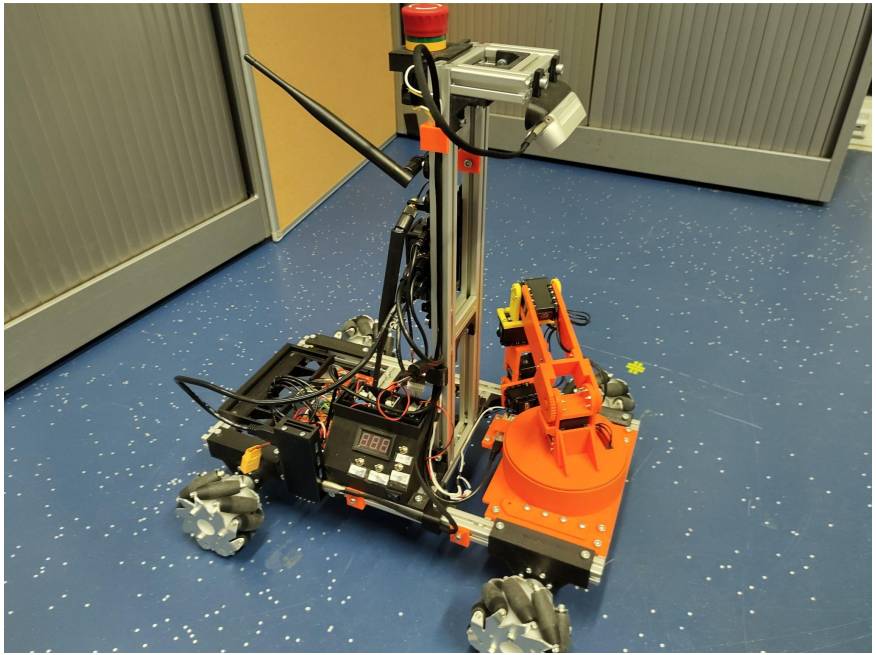
Tutorial: https://www.tensorflow.org/tutorials/images/cnn

Behaviour integration



Platform Instructions

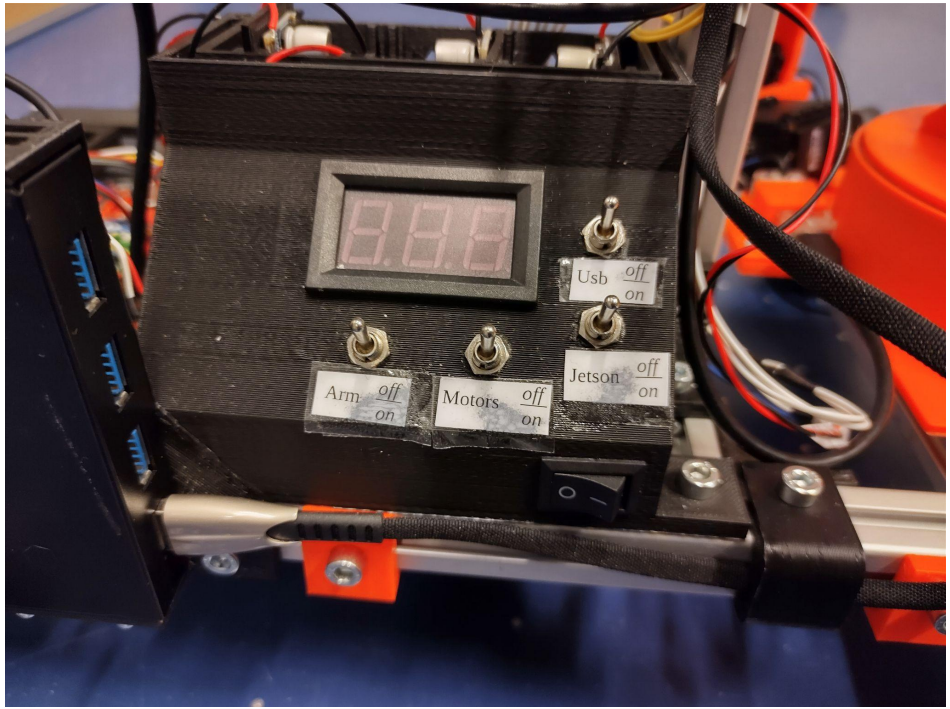
- Turning on the platform
- Running programs on the platform.
- File transfer
- Turning off the platform



Platform Instructions

Power buttons:

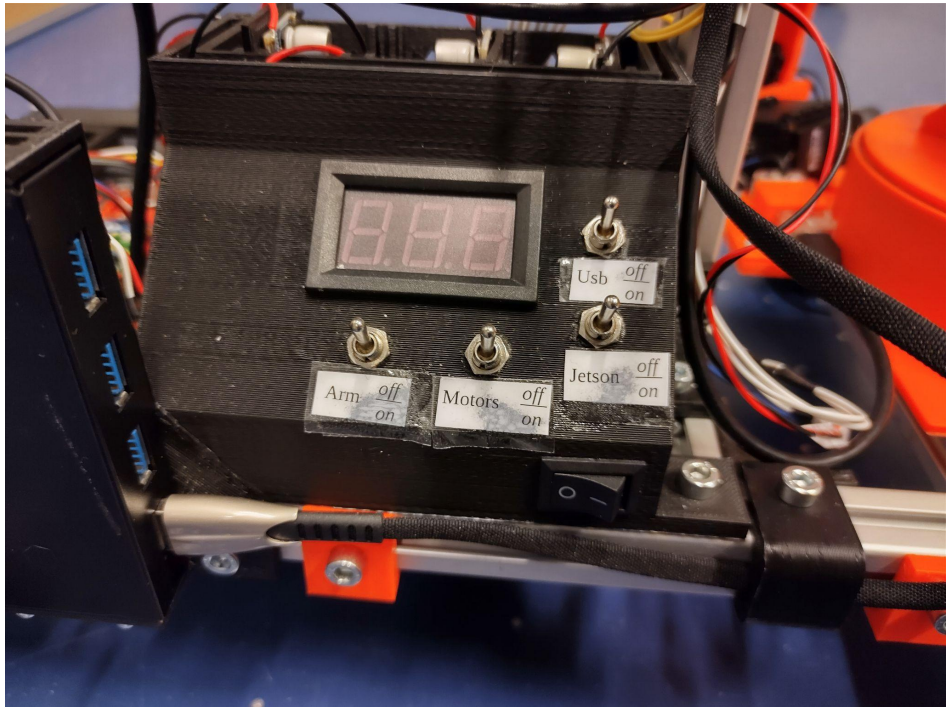
- Main power switch
- Jetson
- USB
- Arm
- Motors



Platform Instructions

Power buttons:

- First set all buttons to *off*.
- Enable the main power switch.
- Up to object recognition:
  - Turn on the Jetson
  - Turn on the USB
  - Turn on the Motors



Platform Instructions

SSH usage:

- ssh-copy-id roboticsX@nano
  - This will make sure that you don’t have to type your password at every SSH login.

- ssh roboticsX@nano
  - This gives you a terminal on the robot

Platform Instructions

Cloning and building

- Clone your repository in catkin_ws/src

- Build your repository with the following command in catkin_ws/src:
  - catkin build -j2

Platform Instructions

ROS

- Run *roscore* on the robots
- To allow a pc terminal to communicate with the robot, use the following command in every terminal that should communicate with the robot, e.g. to use RViz:
  - **export ROS_MASTER_URI=http://nano:11311**

Platform Instructions

Start up

To start all sensors and controllers, run ***on the robot:***

roslaunch robot bringup_robot.launch

To start the arm controller, run ***on the robot:***

roslaunch dynamixel_arm_controller arm_controller.launch

To start navigation, assuming you have the map, run ***on the robot:***

roslaunch navigation real_world.launch

Platform Instructions

To transfer your CNN model to the robot:

scp -r folder_name roboticsX@nano:~/data/

Platform Instructions

To shutdown the robot:

sudo poweroff

Wait until the green LED of the Jetson is off, then power-off the robot


That's all Folks!