Institut de Robòtica
i Informàtica Industrial

# Leg Detector and MHT Tracker tutorial: Docker installation and launch

Ely Repiso, Fernando Herrero, Alberto Sanfeliu.

European Commission | Horizon 2020 European Union funding for Research & Innovation

AI4EU TERRINet The European Robotics Research Infrastructure Network

This work has received partial founding from European Union's Horizon 2020 research and innovation program under the projects AI4EU (H2020-ICT-2018-2-825619) and TERRINet (H2020-INFRAIA-2017-1-two-stage-730994)

CSIC Consejo Superior de Investigaciones Científicas

UPC Universitat Politècnica de Catalunya BarcelonaTech

# 1. Brief Description

Set of C++ libraries with ROS wrapper packages to perform people leg detection and MHT tracking. These packages include a Docker instillation and launch, which is like a virtual machine. Set-up for Ubuntu18.04 with ROS-Melodic. The docker can be used in any Ubuntu system, but the ideal one is Ubuntu18.04, it is to say, maybe some functionalities are reduced in other Ubuntu version.

This installation instructions are also included in the README.md file included inside the docker with name: iri_people_detection_and_tracking_docker_melodic.zip.

With this simulation you can detect and track multiple pedestrians using a 2D lidar.

Developed at Institut de Robòtica i Informàtica Industrial (IRI, CSIC-UPC) (www.iri.upc.edu).

**Contact:** IRI-MobileRoboticsLab support staff, labroboticamobil@iri.upc.edu

**Detector description:**

C++ library [1] to detect people using a Boosting based classifier. The person's leg data is coming from a 2D range lidar. An already trained classifier is provided in /data/classifierData/boostFile.txt, based on a dataset using a Hokuyo UTM-30LX sensor at ~0.4m height from the ground. Also, we included a filter of the false positive that the detector can found in the walls of any map [3].

To use it with ROS, see its ROS wrapper package [2] , allowing online running and showing an example using a recorded sequence.

Laser people detection: uses 2D range data at mid-leg height to detect people.

- [**C++ lib**][**URL**] [1] https://gitlab.iri.upc.edu/labrobotica/algorithms/laser_people_detection
- [**ROS wrapper**][**URL**] [2] https://gitlab.iri.upc.edu/labrobotica/ros/perception/iri_laser_people_detection
- [**Reference**][Spanish] https://upcommons.upc.edu/handle/2099.1/16728

Laser people map filter: uses a OccupancyGrid (ROS map) to filter out people detections located over or near known obstacles.

- [**ROS**][**URL**] [3] https://gitlab.iri.upc.edu/labrobotica/ros/perception/iri_laser_people_map_filter

**Tracker description:**

C++ library [5] to track people based on Kalman filter. It uses 2D points with (x,y) that can come from detections of lidar or other types of sensors. To use it with ROS, see the ROS wrapper package [6] . Also, you can only use the library in C++ to track points, but you have to make an executable program to do it.

The implementation follows a similar approach of the work presented in [Reid, 1979 [7] ], and

some of the contributions presented in [Luber et al, 2011 [8] ]. Our implementation, was published in [Vaquero et al, 2015 [9] ] and [Vaquero et al, 2019 [10] ], and as a final-year degree project and master thesis in [ Repiso, 2015 [11] ].

The people tracker uses a Kalman filter to propagate the pedestrians' trajectories and it combines the different detections with the existing tracks Ids to calculate the most likely association hypothesis between them. The tracker uses hypothesis based on probabilities to confirm, hold, associate and delete the people tracks. Only time-consistent detections that are repeated multiple times become confirmed tracks. The algorithm can handle occlusions and crossings of detections. Furthermore, if we have an environment map, the system can filter the detections situated on a map, avoiding multiple false positive of person's leg detections. The leg detections are 2D points that corresponds to the central position of the person over the 2D plane. Then, the tracker can serve to track any kind of detections represented in 2D coordinates, also can be adapted to track detections in 3D coordinates.

If you use the software, please cite the following ( [10] ) publication.
[10] Vaquero, V., Repiso, E., & Sanfeliu, A. (2019). Robust and real-time detection and tracking of moving objects with minimum 2D LIDAR information to advance autonomous cargo handling in ports. Sensors, 19(1), 107.

This software is usually used with the iri_laser_people_detection. All of the parameters of the tracker are correctly set up for this combination. If you use another type of detector or system, you have to set up the tracking parameters. An explanation of these parameters is included in the ReadMe.txt of the C++ library [5] . Note that in that case, you have also to set up the ROS launch file to use as input the topics of that detector.

Furthermore, we use this tracker and detector with a filter to remove the detections of the laser that are part of a predefined map of the environment. We do this to reduce the overload of the detector, with detections that are not really people. Then, It is recommended to do the same to obtain good results. To do it you can filter the detections that are very near to the map by proximity. Create this node is very easy or also you can ask for the node to the IRI technicians, because this filter node is not included with the iri_people_tracking_mht or iri_laser_people_detection.

Finally, you can find some additional explanations and how to install the packages in the ReadMe.txt included in the C++ library [5] .

People tracking mht: based on Kalman filter

- [**C++ lib**][**URL**] [5] https://gitlab.iri.upc.edu/labrobotica/ros/perception/iri_people_tracking_mht/-/tree/master/tracking_library_trunk
- [**ROS wrapper**][**URL**] [6] https://gitlab.iri.upc.edu/labrobotica/ros/perception/iri_people_tracking_mht
- [**Reference**] [English]
- [7] Reid, D. (1979). An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control, 24(6):843-854.
- [8] Luber, M., Diego Tipaldi, G. & Arras, K. O. (2011). Place-dependent people tracking. The International Journal of Robotics Research, 30(3), 280-293.
- [9] Vaquero, V., Repiso, E., Sanfeliu, A., Vissers, J., & Kwakkernaat, M. (2016). Low cost, robust and real time system for detecting and tracking moving objects to automate cargo handling in port terminals. In Robot 2015: Second Iberian Robotics Conference (pp. 491-502). Springer, Cham.
- [10] Vaquero, V., Repiso, E., & Sanfeliu, A. (2019). Robust and real-time detection and tracking of moving objects with minimum 2D LIDAR information to advance autonomous cargo handling in ports. Sensors, 19(1), 107.
- [11]https://upcommons.upc.edu/bitstream/handle/2117/84172/Robust_Multi_Hypothesis_Tracker_Fusing_Diverse_Sensor_Inform.pdf

# 2. Docker Installation

**2.1. Install the docker:** Open a terminal on Ubuntu.

**/home$** sudo apt install docker.io

**/home$** sudo systemctl start docker

**/home$** sudo systemctl enable docker

**2.2. Install plugin-docker:**

**/home$** distribution=$(. /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

**/home$** curl -s -L https://nvidia.github.io/nvidia-container-runtime/experimental/$distribution/nvidia-container-runtime.list | sudo tee /etc/apt/sources.list.d/nvidia-container-runtime.list

**/home$** sudo apt-get update

**/home$** sudo apt-get install -y nvidia-docker2

**/home$** sudo apt-get update

**/home$** sudo apt-get upgrade

**2.3. Enable docker run without need of sudo:**

**/home$** sudo usermod -aG docker ${USER}

**/home$** su - ${USER}

**/home$** sudo systemctl restart docker

# 3. Create the Docker Image and Compile the system

Open a terminal on Ubuntu.

**/home$** cd /path_to_Dockerfile  // arrive to the docker folder.

**/home$** docker build -t iri_people_detection_and_tracking_docker_melodic .

# 4. How to launch the system

**4.1. Run container from created image:** Open a terminal on Ubuntu.

**/home$** cd /path_to_Dockerfile

**/home$** ./pal_docker.sh -it iri_people_detection_and_tracking_docker_melodic /bin/bash  # creates a new container

**4.2. Inside the Docker image (virtual machine), launch gazebo detection and tracking nodes for Dabo-robot in BRL:** Same terminal than before.

**/home$** roslaunch iri_dabo_gazebo sim_gazebo_dabo_people_detection_and_tracking.launch

**Note:** This launch, launchs also the Rviz visualization. In Rviz you can visualize the scene with the map, robot position, person detections, laserscan data, etc. Also, launchs the rqt_reconfigure to be able to dynamically change parameters.

**4.3 If you want to move any person of the environment.**

You need to: Open a new terminal inside the container and run a person teleoperation node.

**/home$** cd /path/to/Dockerfile

**/home$** docker exec -it $(docker ps -l -q) bash

**/home$** rosrun teleop_twist_keyboard teleop_twist_keyboard.py cmd_vel:=/person1/cmd_vel __name:=person1

**Note:** The last command will allow you to teleoperate `person1`. Modify to teleoperate another available person on the scene.

**4.4. Brief description of the simulation markers:**

If you want to see the markers in rviz corresponding to that data, you have to habilitate the flags in the rviz corresponding to the detections, the filtered-detections and the tracks.

1. Markers of the people detections, represented in blue: MarkerArray=>/lpd_2/markers.

2. Markers of the filtered people detections, represented in yellow: MarkerArray=>/laser_people_map_filter/markers.

3. Markers of the people tracks, represented in green and Markers of associated detections represented in red: MarkerArray=>/mht_2/tracksMarkers.