

# Applications of Heterogeneous Smart Robots in Modern Industry



Anass Saudi, Ahmed El-Sayed  
Department of Electrical and Computer Engineering  
University of Bridgeport, Bridgeport, CT

## Abstract

Industry 4.0 pushes the concept of flexible manufacturing, which means the ability to adapt to the frequent demand customization of low/high-volume production. However, this kind of production is not mostly automated, and therefore, it requires human intervention and interaction. In order to achieve that goal, robots should exhibit autonomous and intelligent functionality to adapt to the uncertainty of the environments and target changes. This work utilizes the usage of ROS (robot operating system) services as a management tool to coordinate the collaboration of different robot types for a predefined task in the industrial field. This work proposes a structured ROS approach for real-time control of heterogeneous robots. This structure provides a network of robots connected to one server (Master station), which can send & receive information to multiple robots (Nodes/Slaves) of different types. Using the proposed technique, robots will actively interact with the working environment and perform multiple tasks safely in collaboration of human workers. This solution provides a smart industrial approach, where linking two different processes/factories will be safer and faster.

## Proposed System

- ❑ The proposed solution creates a single large ROS network managed by a single roscore master, which can connect multiple nodes to exchange information/topics as stated in Fig. 1.
- ❑ Multiple robotic arms, nodes for ROS, will be connected to one ROS mastercore through a shared server (Fig. 2).
- ❑ The master are running on the IP1 address.
- ❑ Arm 1 on IP2 address and arm 2 on IP3 address will be running as nodes.
- ❑ Industrial cameras, sensors, and actuators' status will be considered ROS topics on the network.
- ❑ The master could also run some other ROS nodes.

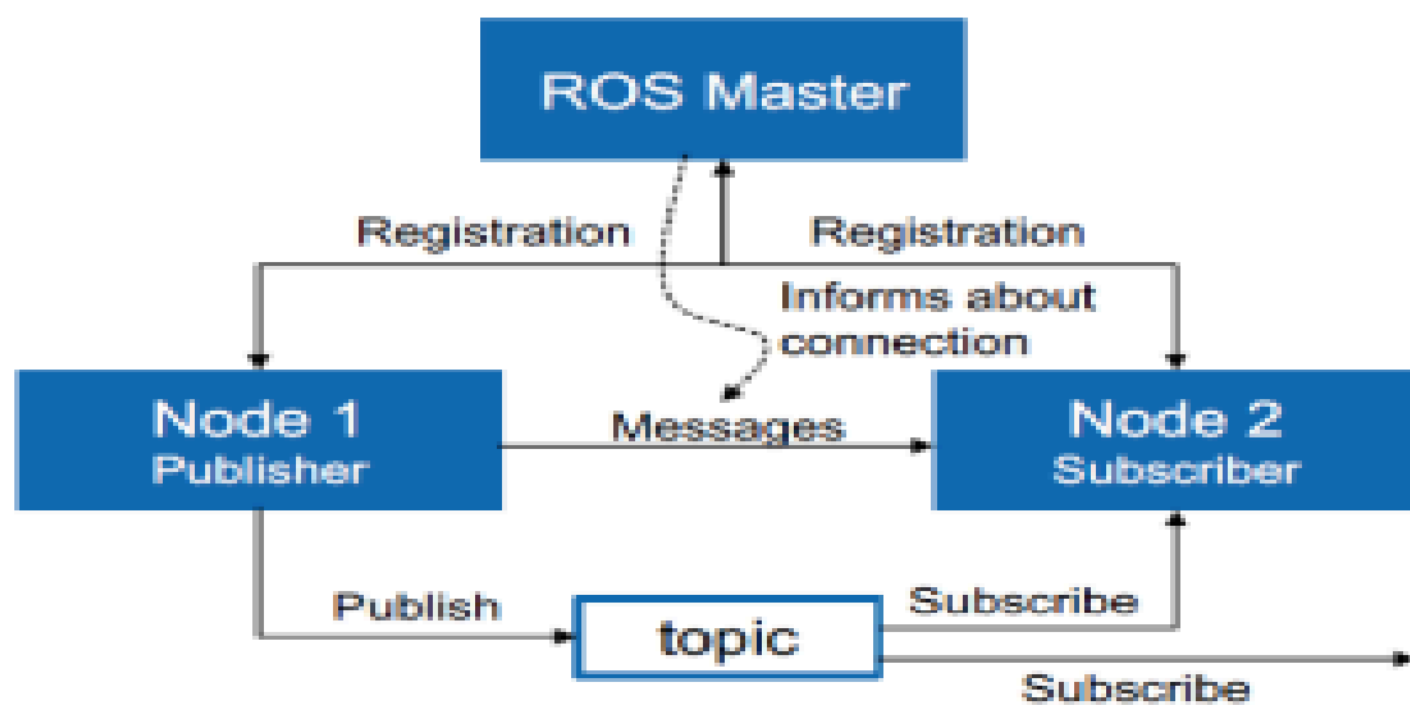


Fig. 1: ROS master, nodes & topics hierarchy.

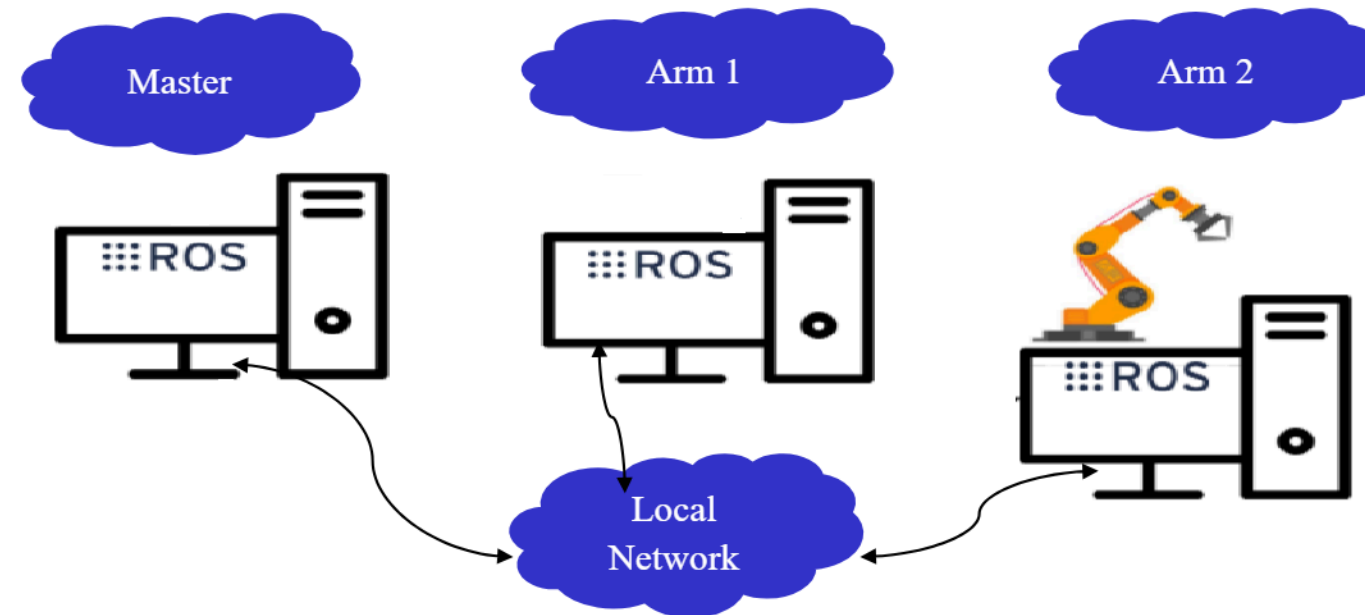


Fig. 2: ROS master/slave network

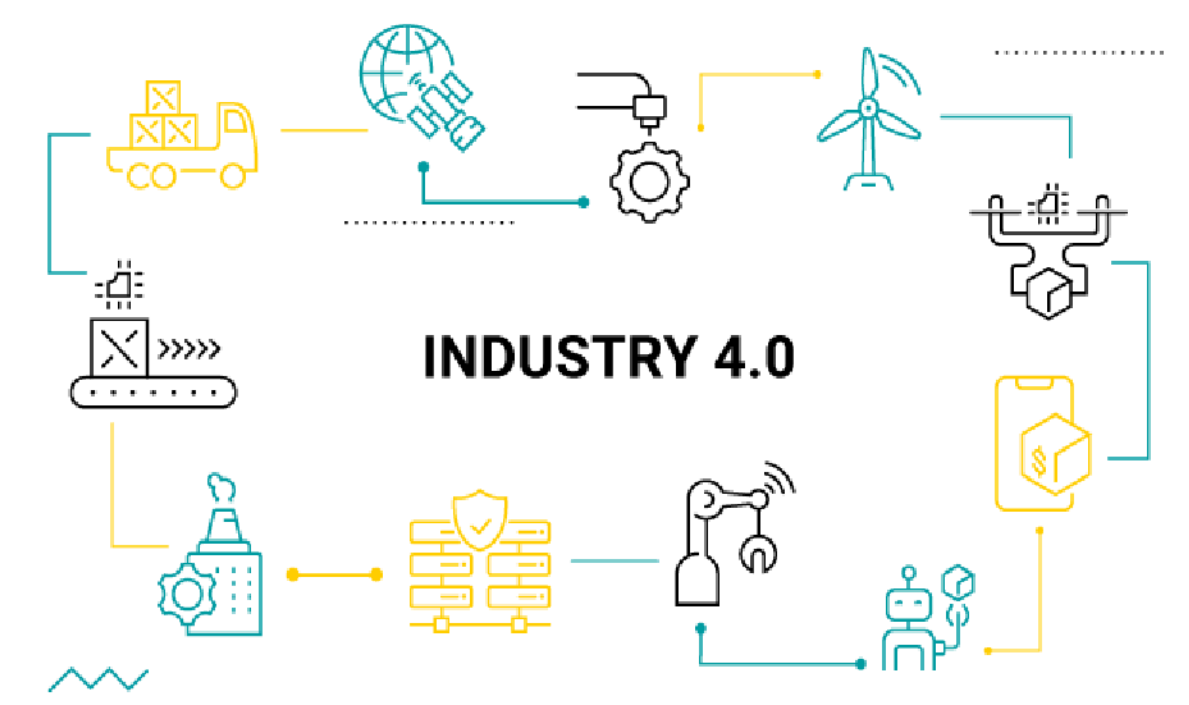


Fig. 3: Industry 4.0 (future) network

## Results

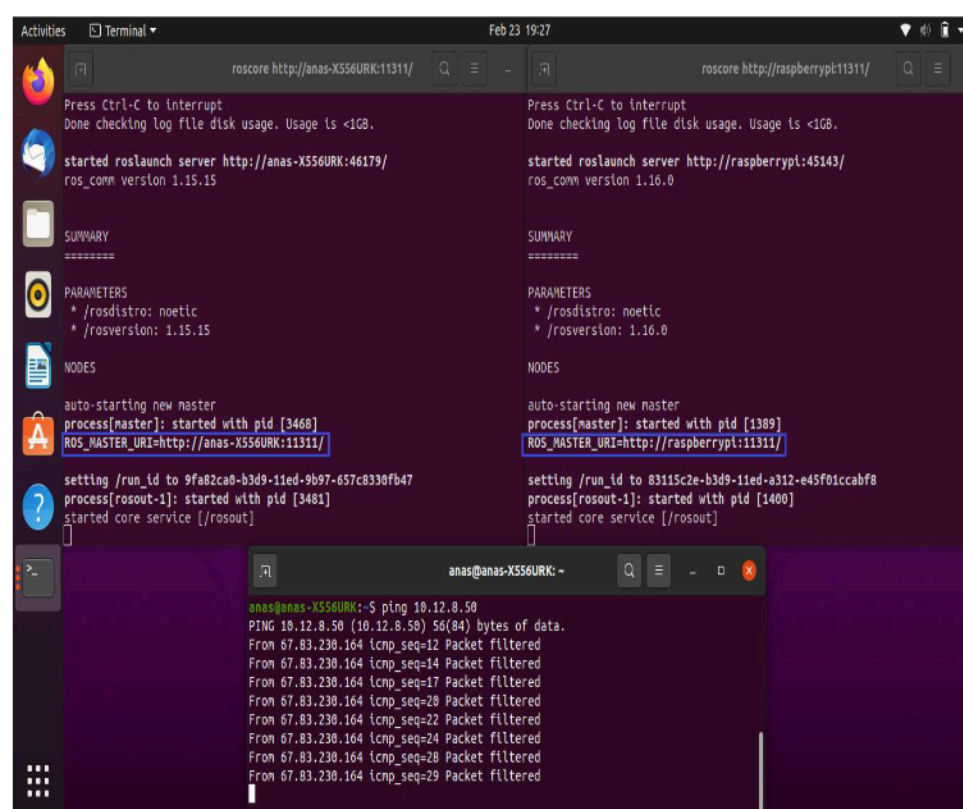


Fig. 4: Two masters terminals

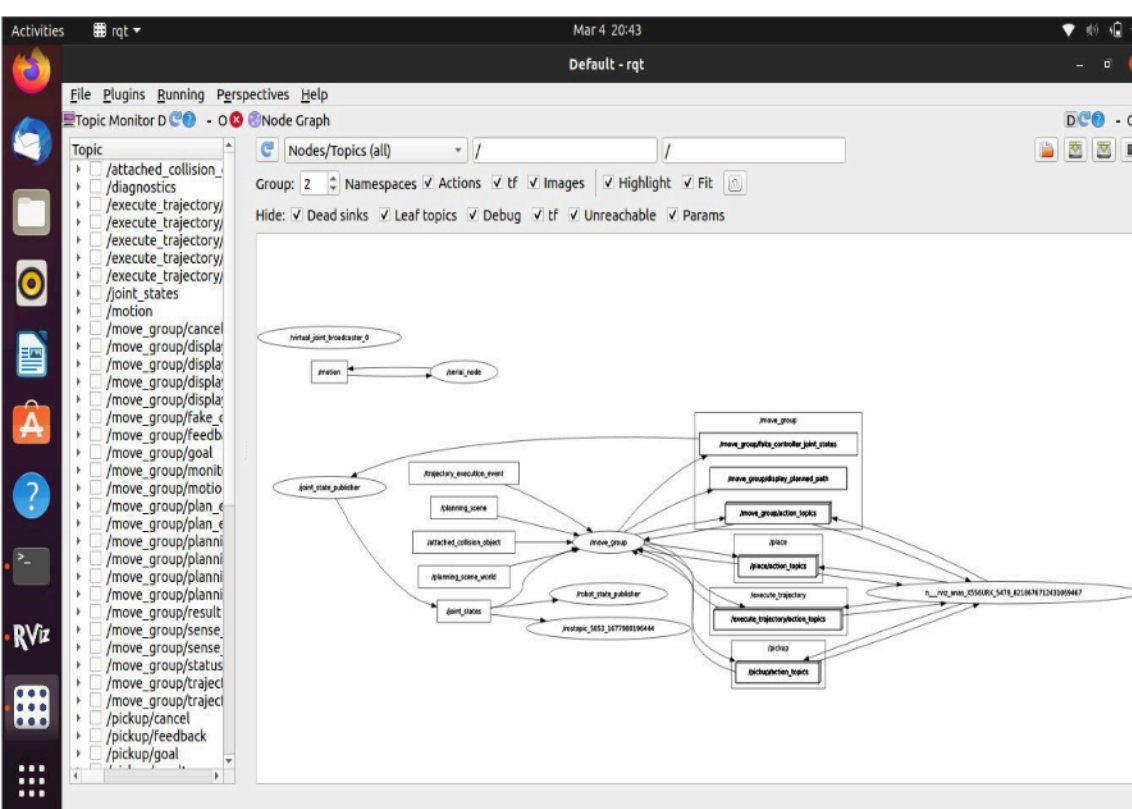


Fig. 5: Topics graph from two nodes

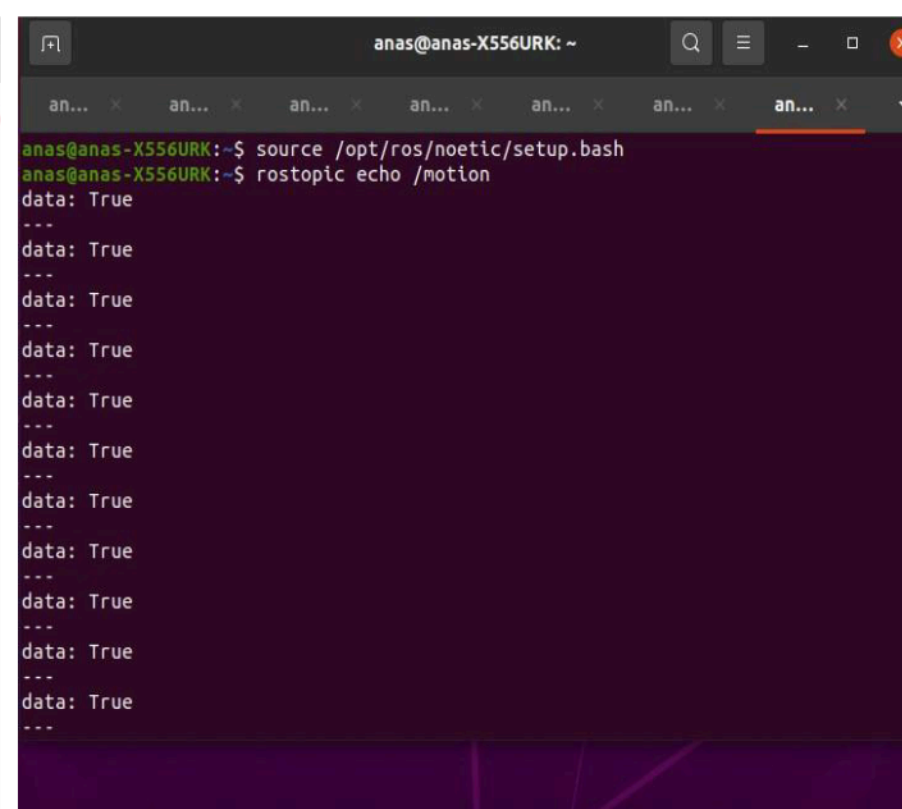


Fig. 6: Proximity sensor status

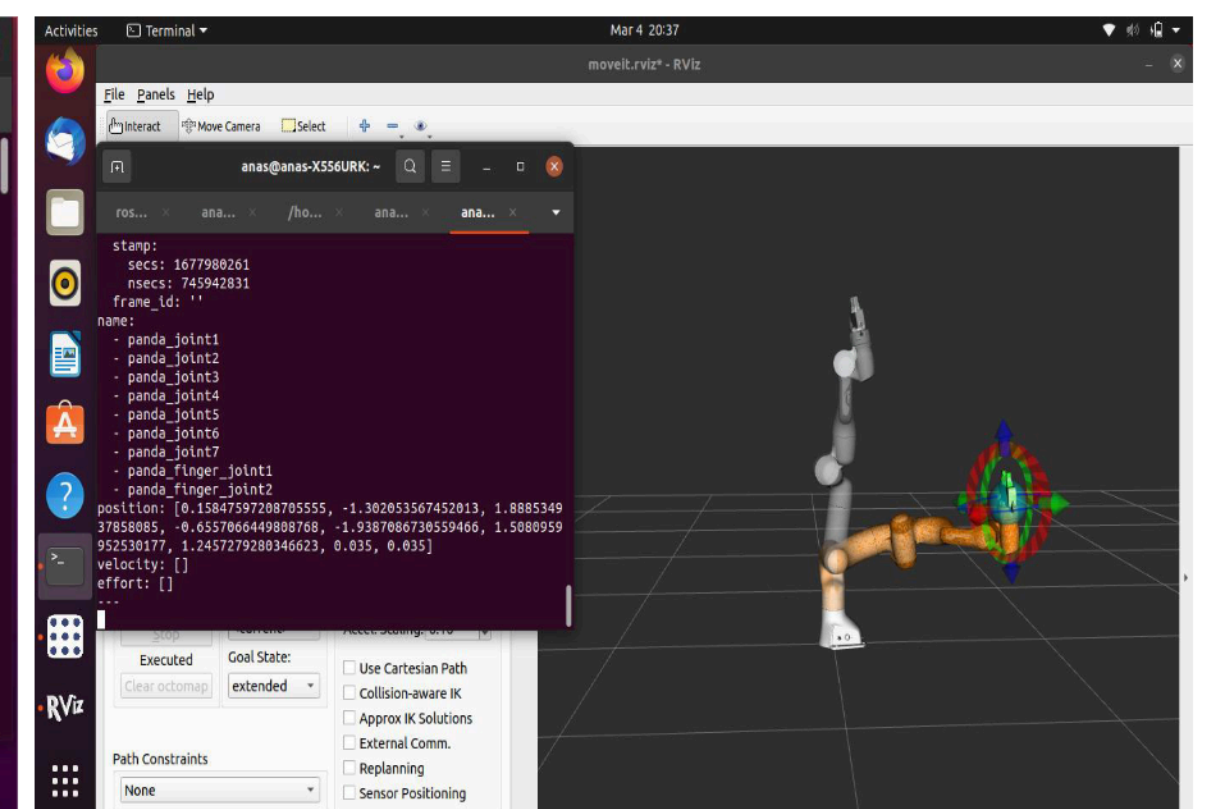
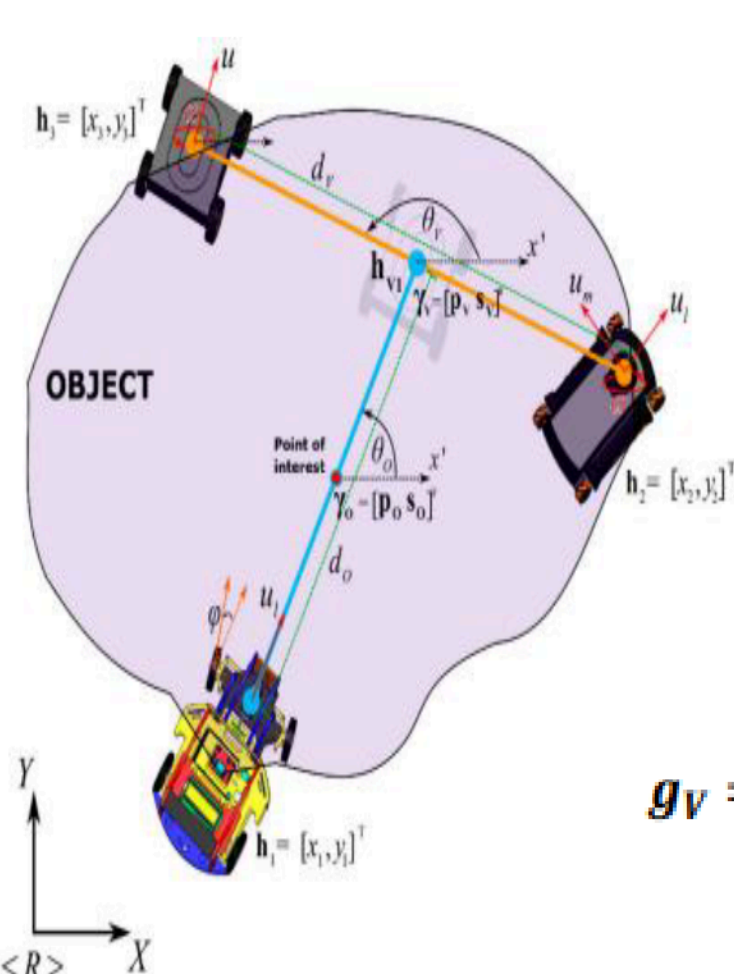


Fig. 7: Arm motion planning on RVIZ

- ❑ Fig 4 shows two ROS masters running on the same server from two different machines, Raspberry pi 4 and Linux station.
- ❑ Fig.5 shows all the topics running from two nodes.
- ❑ Fig.6 shows the terminal, where we can publish or subscribe to any of the topics of the node, like reading the proximity sensor topic.
- ❑ Fig.7 shows a ROS arm running on Rviz, where we can simulate different motion planning scenes of the arm, also reading the position in real-time.
- ❑ The results presented in this poster present several advantages: simplicity of use, since the user only has to select which information to share, and periodic updates, which keep the local network synchronized with the global network information.

## Conclusions & Future work

After setting up the Robotic Arm network, we are modeling a network to control a group of ground heterogeneous robots that can carry a certain load, where more than one robot can be connected to one ROS master in order to achieve a specific task load, and always one of them is in the standby state, ready to receive data from the Robotic Arms master, where it can be used to link high-distance processes.



$$\begin{aligned}
 \gamma_v &= \begin{bmatrix} p_v \\ s_v \end{bmatrix}; p_v = \begin{bmatrix} x_v \\ y_v \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(x_3 + x_2) \\ \frac{1}{2}(y_3 + y_2) \end{bmatrix}; s_v = \begin{bmatrix} d_v \\ \theta_v \end{bmatrix} = \begin{bmatrix} \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2} \\ \tan^{-1}\left(\frac{y_3 - y_2}{x_3 - x_2}\right) \end{bmatrix} \\
 \gamma_o &= \begin{bmatrix} p_o \\ s_o \end{bmatrix}; p_o = \begin{bmatrix} x_o \\ y_o \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(x_v + x_1) \\ \frac{1}{2}(y_v + y_1) \end{bmatrix}; s_o = \begin{bmatrix} d_o \\ \theta_o \end{bmatrix} = \begin{bmatrix} \sqrt{(x_v - x_1)^2 + (y_v - y_1)^2} \\ \tan^{-1}\left(\frac{y_v - y_1}{x_v - x_1}\right) \end{bmatrix} \\
 g_v &= \begin{bmatrix} h_3 \\ h_2 \end{bmatrix} = \begin{bmatrix} x_v + \frac{1}{2}d_v \cos \theta_v \\ y_v + \frac{1}{2}d_v \sin \theta_v \\ x_v - \frac{1}{2}d_v \cos \theta_v \\ y_v - \frac{1}{2}d_v \sin \theta_v \end{bmatrix}; g_o = \begin{bmatrix} p_v \\ h_1 \end{bmatrix} = \begin{bmatrix} x_o + \frac{1}{2}d_o \cos \theta_o \\ y_o + \frac{1}{2}d_o \sin \theta_o \\ x_o - \frac{1}{2}d_o \cos \theta_o \\ y_o - \frac{1}{2}d_o \sin \theta_o \end{bmatrix}
 \end{aligned}$$

## References

1. <http://wiki.ros.org/Papers>
2. <https://rosindustrial.org/>
3. <http://wiki.ros.org/ROS/Tutorials/MultipleMachines>
4. <http://wiki.ros.org/roscerial>
5. *ROS for Robotics Programming; Lentin Joseph | Jonathan Cacase.*
6. *Multirobot Heterogeneous Control Considering Secondary Objectives Mastering; Julio F. Acosta | Javier Garrido.*