





# ROS Tutorial

Software version 1.86

Training Center

Nov 2022

本文所有資訊屬於達明機器人（股）公司（以下簡稱本公司）財產，未經本公司事先授權不得以任何形式或方式轉載及複製任何資料。本文任何資訊不應視為任何要約或是承諾，日後如有變更，恕不另行通知。本說明書應定期審查，本公司不會對本文任何錯誤或是遺漏承擔責任。

 和  標誌為達明機器人（股）公司註冊商標，本公司保留本說明書及其拷貝的所有權及其著作權。

 達明機器人股份有限公司  
TECHMAN ROBOT INC.



# ROS教學

TECHMAN ROBOT-TRAINING CENTER

## 目錄

### 01. ROS 介紹

ROS 基本介紹  
ROS版本  
ROS常用Library

### 04.ROS Vision

ROS Vision dependencies  
ROS Vision 使用說明

### 02. TM ROS Driver環境建置

Melodic  
Dashing

### 05. Moveit

Moveit 介紹  
Moveit 使用說明

### 03. TM ROS Driver 連線

Ethernet slave setting  
Network setting  
Listen node  
ROS1 連線  
ROS2 連線

### 06. Demo Code 使用



# 01.

## ROS介紹

- ROS基本介紹
- ROS版本
- ROS常用Library

## ROS 基本介紹

ROS

1. Robot operating system的縮寫
2. 主要在Linux上 (最新版的support Windows)
3. 優點: 很多第三方的library
4. 缺點: 因為第一版在2007年就開發完成  
因此有一些過時的技術包袱

ROS 2

1. Robot operating system 2 (ROS2)
2. 主要在Linux上, 也有支援Windows 和 Mac
3. 優點: 改善ROS的過時技術
4. 缺點: 第三方的library還不完善

# ROS 版本

ROS 1 Distribution Releases <sup>[52]</sup>

Distribution	Release date	Poster	EOL date	Support duration
Noetic Ninjemys (last ROS 1 release)	23 May 2020		May 2025	5 years
Melodic Morenia	23 May 2018		2023-05-30	5 years
Lunar Loggerhead	23 May 2017		2019-05-30	2 years
Kinetic Kame	23 May 2016		2021-05-30	5 years

ROS 2 Distribution Releases <sup>[53][1]</sup>

Distribution	Release date	Poster	EOL date	Support duration
Rolling Release <sup>[2][3]</sup> (rolling release with latest features)	progressing since June 2020	(unknown)	N/A	N/A
(J Turtle)	May 2024	t.b.d.	EST. May 2029	5 years
(I Turtle)	May 2023	t.b.d.	EST. November 2024	1.5 years
Humble Hawkbill	May 2022	t.b.d.	EST. May 2027	5 years
Galactic Geochelone	23 May 2021 <sup>[4]</sup>		November 2022	1.5 years
Foxy Fitzroy	5 June 2020 <sup>[5]</sup>		May 2023	2 years
Eloquent Elusor	22 November 2019		November 2020	1 year
Dashing Diademata	31 May 2019		May 2021	2 years

有圈起來的是我們目前有支援(2022)

# ROS 版本

ROS和Linux版本對應的關係

Ubuntu版本	ROS1	ROS2
18.04	Melodic	Dashing
20.04	Noetic	Foxy
22.04	ROS1不再支援	Humble

## ROS 常用Library



Moveit 是一個軌跡規劃和避障的軟體  
最近有越來越多的使用者在使用

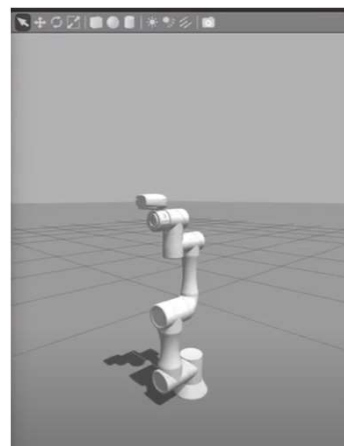
	Moveit1	Moveit2
特點	普通的模型	可以撈手臂的資料 讓模型跟實際相同
製作時間	早	晚
主要位置	ROS1(已經下架)	ROS1(剛上架)ROS2

## ROS 常用Library



GAZEBO

Gazebo是一個有物理引擎的模擬器  
預計明年初之前會開始支援



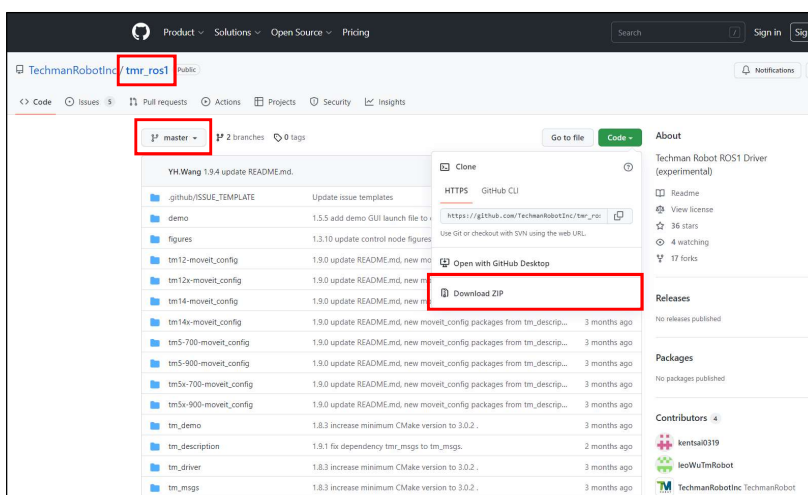
# 02.

## TM ROS Driver 環境建置

- TM ROS Driver 環境建置(Melodic)
- TM ROS Driver 環境建置(Dashing)

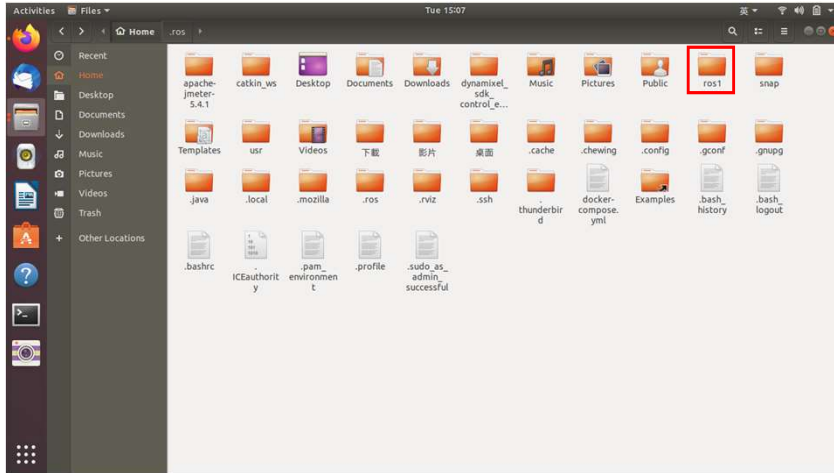


## TM ROS Driver (Melodic)



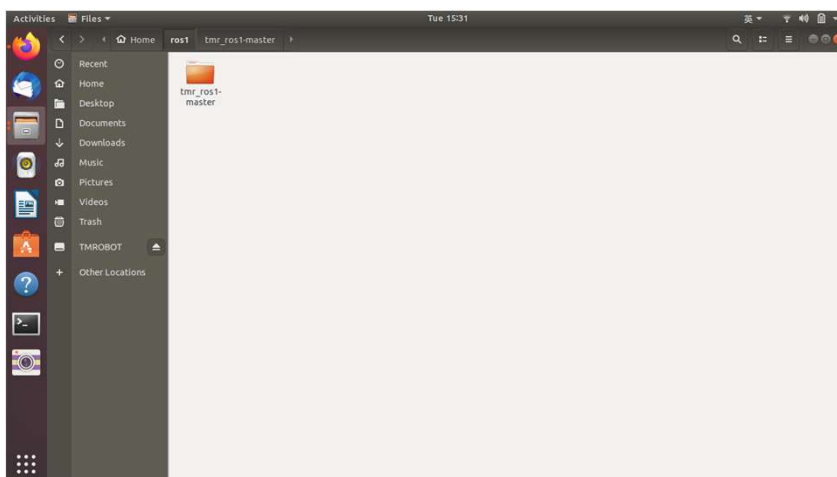
Step1  
到 TechmanRobotInc的Github  
上下載正確版本的TM Ros Driver  
<https://github.com/TechmanRobotInc>

## TM ROS Driver (Melodic)



Step2  
在Linux系統的Home下面新增一個新資料夾

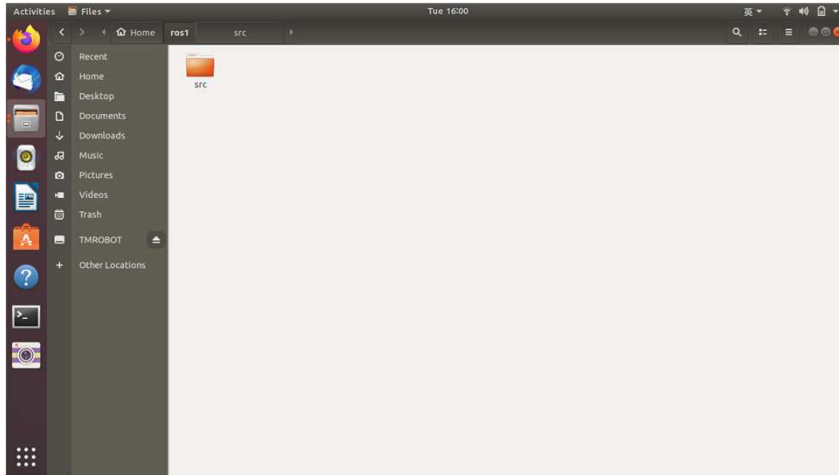
## TM ROS Driver (Melodic)



Step3  
將下載下來的ROS Driver解壓縮並放置在Step2建立好的資料夾內

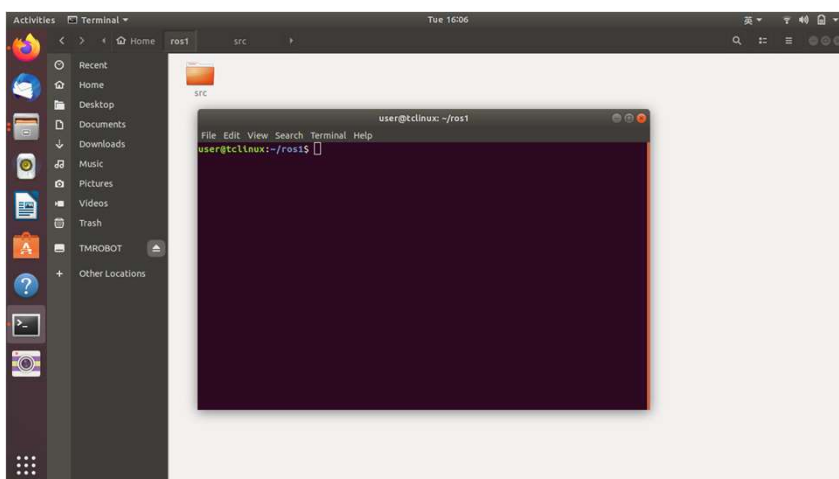


## TM ROS Driver (Melodic)



Step4  
將資料夾更改為 src

## TM ROS Driver (Melodic)



Step5  
在資料夾內按下滑鼠右鍵 · 選擇  
Open in Terminal

## TM ROS Driver (Melodic)

```

user@tclinux: ~/ros1
File Edit View Search Terminal Help
user@tclinux:~/ros1$ source /opt/ros/melodic/setup.bash
user@tclinux:~/ros1$ catkin_make

```

Step6  
在Terminal內輸入

```

source /opt/ros/melodic/setup.bash
catkin_make

```

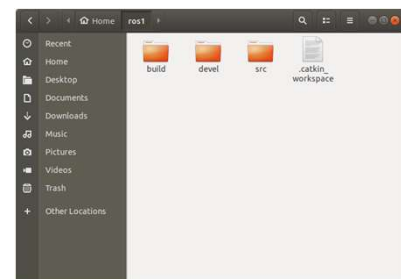
## TM ROS Driver (Melodic)

```

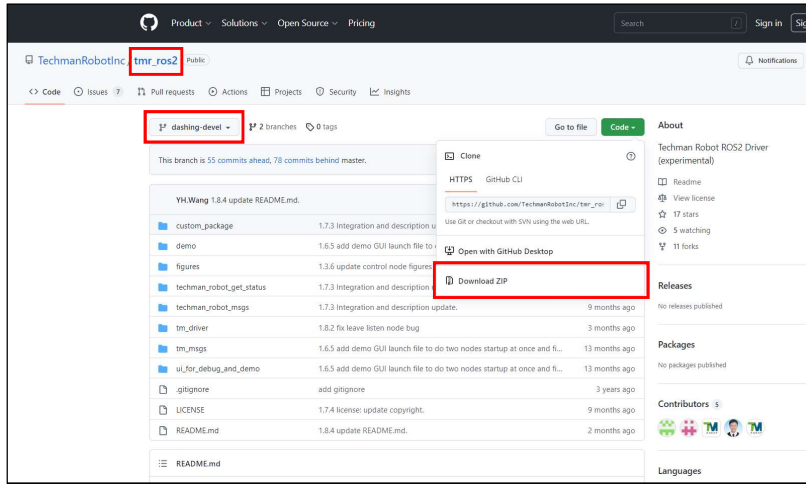
user@tclinux: ~/ros1
File Edit View Search Terminal Help
catkin_make
[ 90%] Building CXX object ui_for_debug_and_demo/CMakeFiles/robot_ui.dir/src/tm_
ros_driver_windows.cpp.o
[ 91%] Building CXX object tm_driver/CMakeFiles/tm_driver.dir/src/tm_communicati
on.cpp.o
[ 92%] Building CXX object tm_driver/CMakeFiles/tm_driver.dir/src/tm_robot_state
.cpp.o
[ 93%] Building CXX object tm_driver/CMakeFiles/tm_driver.dir/src/tm_packet.cpp.
o
[ 94%] Building CXX object tm_driver/CMakeFiles/tm_driver.dir/src/tm_print.cpp.o
[ 94%] Building CXX object tm_driver/CMakeFiles/tm_driver.dir/src/tm_driver_util
ities.cpp.o
[ 95%] Building CXX object ui_for_debug_and_demo/CMakeFiles/robot_ui.dir/robot_u
i_autogen/mocs_compilation.cpp.o
[ 96%] Building CXX object tm_driver/CMakeFiles/tm_driver.dir/src/tm_ethernet_sl
ave_connect.cpp.o
[ 97%] Building CXX object tm_driver/CMakeFiles/tm_driver.dir/src/tm_listen_node
_connect.cpp.o
[ 98%] Linking CXX executable /home/user/ros1/devel/lib/tm_driver/tm_driver
[100%] Linking CXX executable /home/user/ros1/devel/lib/ui_for_debug_and_demo/ro
bot_ui
[100%] Built target tm_driver
[100%] Built target robot_ui
user@tclinux:~/ros1$

```

Step7  
建置完成後，環境內會出現其他  
的資料夾

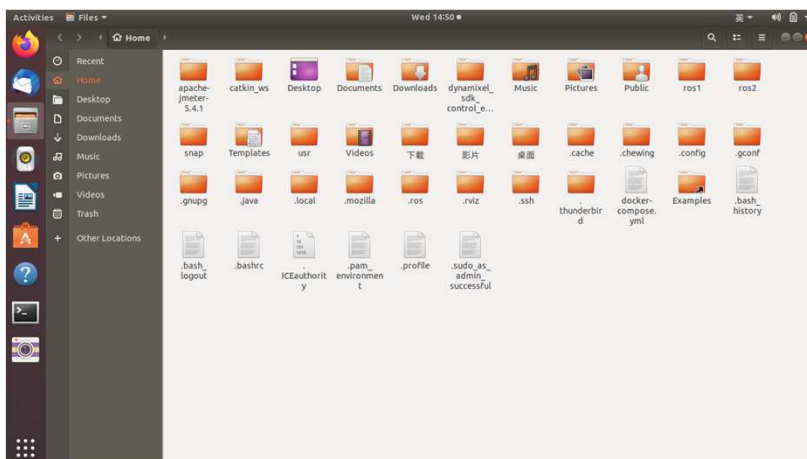


# TM ROS Driver (Dashing)



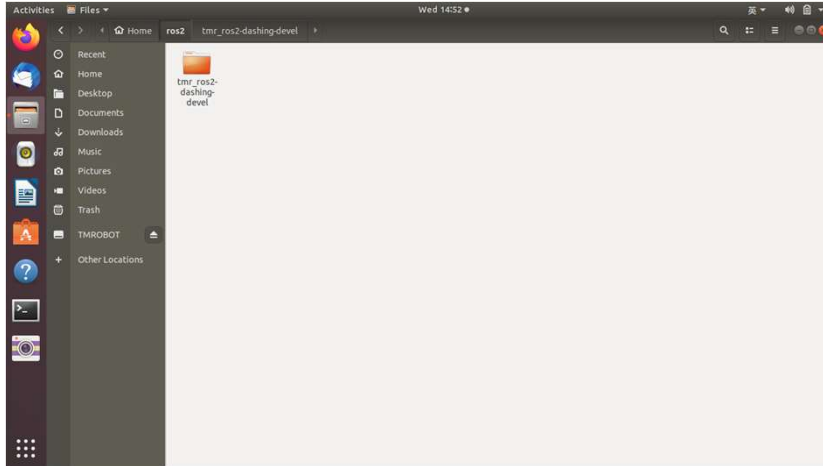
Step1  
到 TechmanRobotInc的Github  
上下載正確版本的TM Ros Driver

# TM ROS Driver (Dashing)



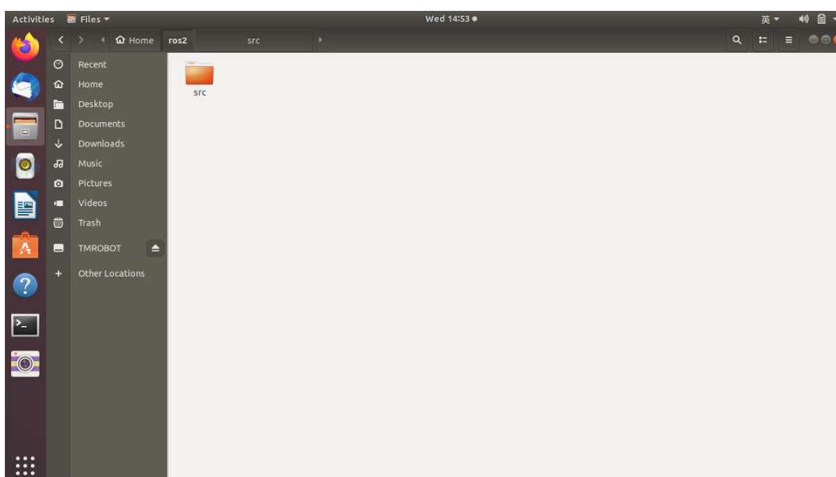
Step2  
在Linux系統的Home下面新增一  
個新資料夾

## TM ROS Driver (Dashing)



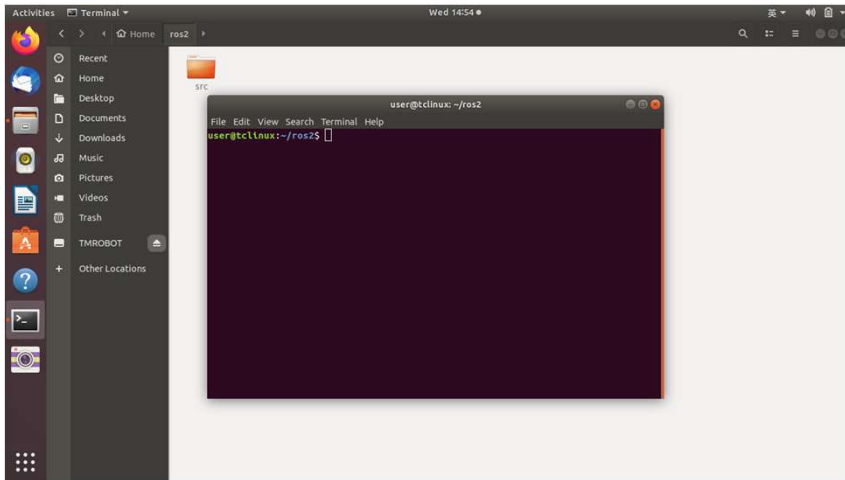
**Step3**  
將下載下來的ROS Driver解壓縮  
並放置在Step2建立好的資料夾內

## TM ROS Driver (Dashing)



**Step4**  
將其名稱更改為 src

## TM ROS Driver (Dashing)



Step5  
在資料夾內按下滑鼠右鍵 · 選擇  
Open in Terminal

## TM ROS Driver (Dashing)

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
user@tclinux:~/ros2$ source /opt/ros/dashing/setup.bash
ROS_DISTRO was set to 'melodic' before. Please make sure that the environment do
es not mix paths from different distributions.
user@tclinux:~/ros2$ colcon build

```

Step6  
在Terminal內輸入

`source /opt/ros/dashing/setup.bash`  
`colcon build`

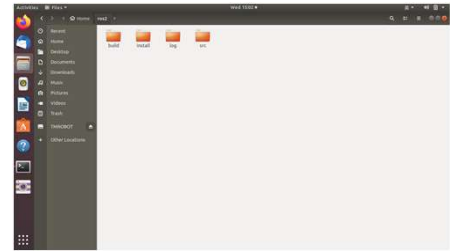
## TM ROS Driver (Dashing)

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
/home/user/.local/lib/python3.6/site-packages/setuptools/dist.py:726: UserWarning: Usage of dash-separated 'install-scripts' will not be supported in future versions. Please use the underscore name 'install_scripts' instead
  % (opt, underscore_opt)
/home/user/.local/lib/python3.6/site-packages/setuptools/command/install.py:37: SetuptoolsDeprecationWarning: setup.py install is deprecated. Use build and pip and other standards-based tools.
  setuptools.SetuptoolsDeprecationWarning,
---
Finished <<< tm_get_status [1.77s]
Finished <<< techman_robot_msgs [19.6s]
Starting >>> custom_package
Finished <<< custom_package [12.6s]
Finished <<< tm_msgs [37.6s]
Starting >>> demo
Starting >>> tm_driver
Starting >>> ui_for_debug_and_demo
Finished <<< ui_for_debug_and_demo [27.8s]
Finished <<< tm_driver [35.9s]
Finished <<< demo [38.2s]

Summary: 7 packages finished [1min 17s]
1 package had stderr output: tm_get_status
user@tclinux:~/ros2$
  
```

Step7  
建置完成後，環境內會出現其他的資料夾

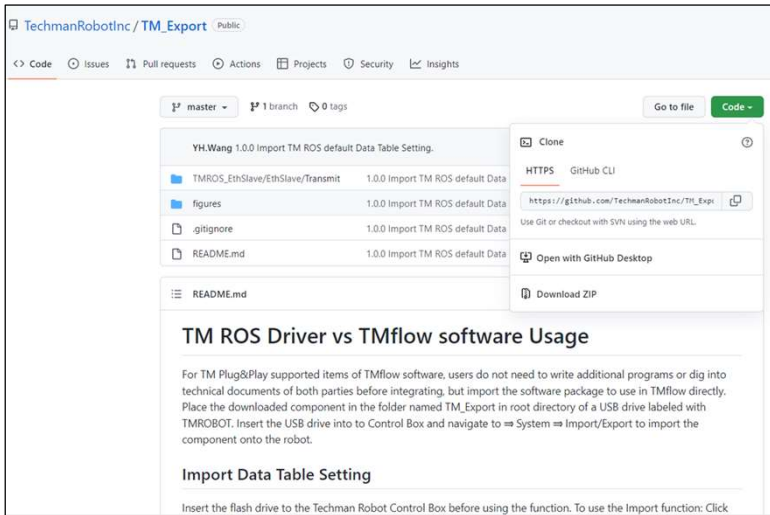


# 03.

## TM ROS Driver 連線

- Ethernet slave setting
- Network setting
- Listen Node
- ROS1 連線(Melodic)
- ROS2 連線(Dashing)

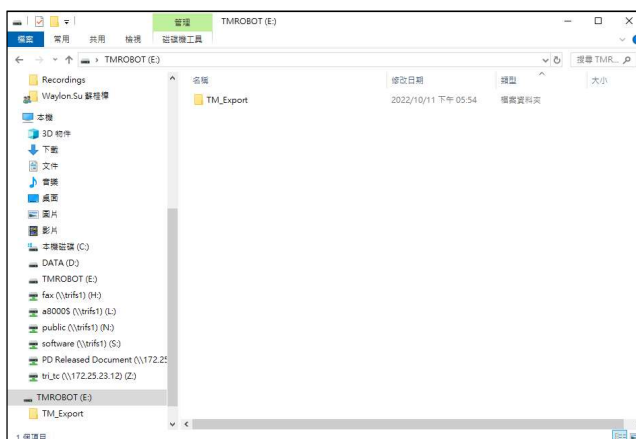
# Ethernet Slave setting



Step1

到Github上下载TM\_Export  
(此为Ethernet slave设定档)

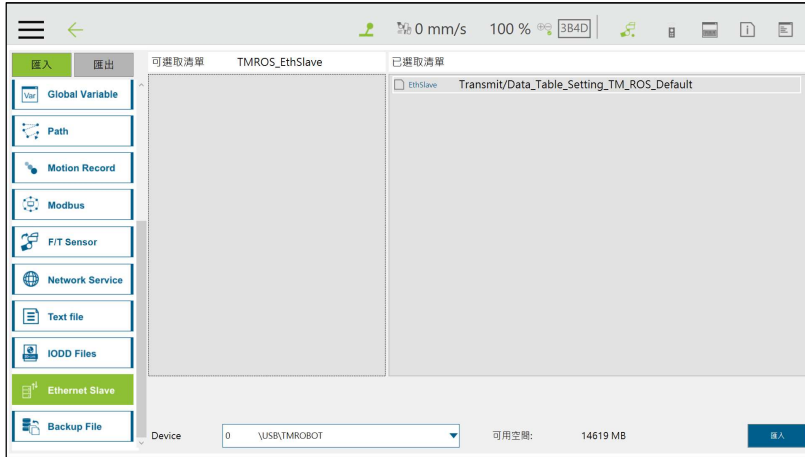
# Ethernet Slave setting



Step2

解壓縮後·將TM\_Export放置於  
TMROBOT隨身碟內

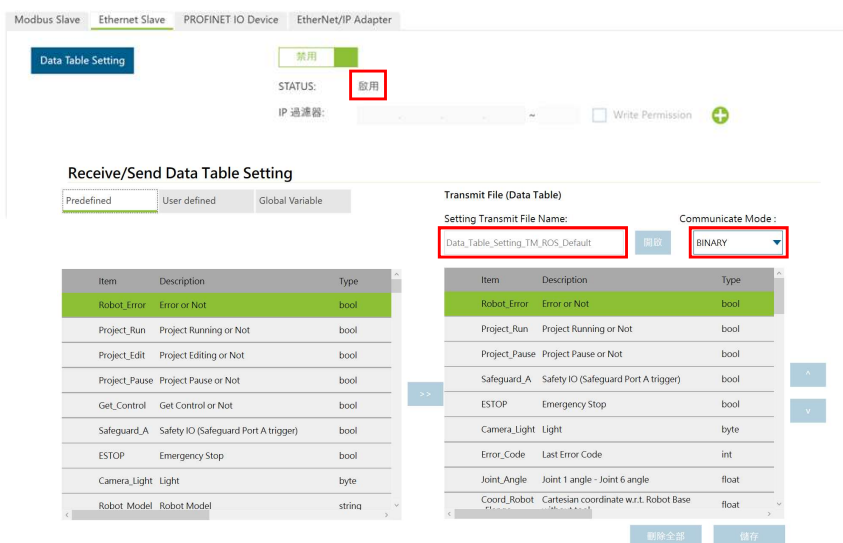
# Ethernet Slave setting



Step3

將下載下來的Ethernet slave設定檔匯入手臂中

# Ethernet Slave setting



Step4

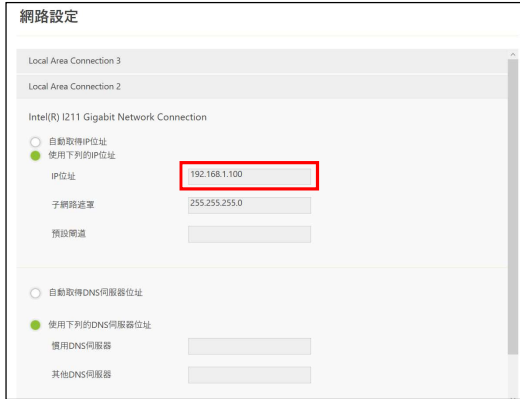
到connection中設定正確的Ethernet Slave setting

通訊模式要選擇Binary

並確定Status為啟用



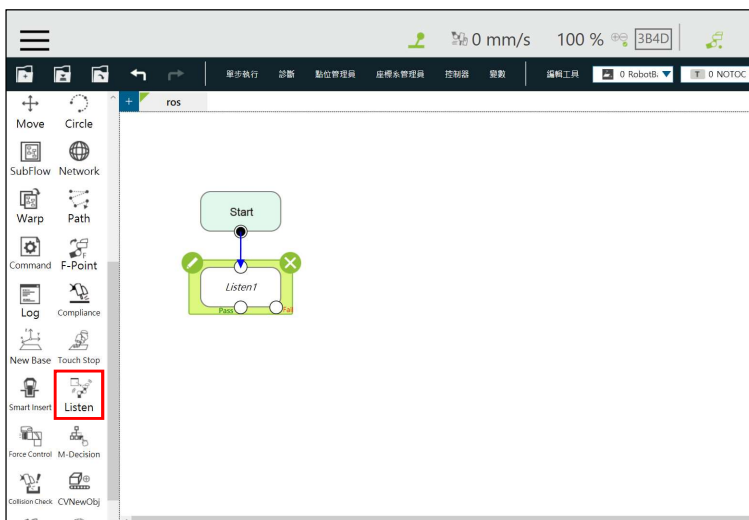
# Network setting



Step5

使用網路線連接電控箱與Linux主機  
並確定他們的IP位置在同一個網域下

# Listen node



Step6

建立新專案並拉出一個Listen node  
然後執行專案

## ROS1 連線(Melodic)

```

roscore http://tclinux:11311/
File Edit View Search Terminal Help
user@tclinux:~$ roscore
... logging to /home/user/.ros/log/07b78c58-4942-11ed-bbcf-f48c5093cb9b/roslauch
h-tclinux-12571.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://tclinux:43799/
ros_comm version 1.14.13

SUMMARY
=====
PARAMETERS
* /rostdistro: melodic
* /rosversion: 1.14.13

NODES
auto-starting new master
process[master]: started with pid [12581]
ROS_MASTER_URI=http://tclinux:11311/

```

Step1

Terminal 1 :  
輸入  
roscore

ROS1需要一個roscore來進行溝通

## ROS1 連線(Melodic)

```

user@tclinux: ~/ros1
File Edit View Search Terminal Help
user@tclinux:~/ros1$ source devel/setup.bash
user@tclinux:~/ros1$ rosrn tm_driver tm_driver 192.168.1.100
[ INFO] [1665481222.834316084]: TM_ROS: Robot_ip:= 192.168.1.100
[DEBUG] TmCommunication::TmCommunication
[DEBUG] TmSBuffer::TmSBuffer
[DEBUG] TmCommRecv::TmCommRecv
[DEBUG] TmRobotState::TmRobotState
[DEBUG] Create DataTable
[INFO] Ethernet slave communication: TmSvrCommunication
[DEBUG] TmCommunication::TmCommunication
[DEBUG] TmSBuffer::TmSBuffer
[DEBUG] TmCommRecv::TmCommRecv
[INFO] Listen node communication: TmSctCommunication
[INFO] Listen node communication: start
[INFO_ONCE]TM_COM: ip:=192.168.1.100
[DEBUG] TM_COM: rv:=0
[DEBUG] TM_COM: Connection is ok
[DEBUG] TM_COM(listen node communication): 0_NONBLOCK connection is ok
[INFO] TM_COM(listen node communication): TM robot is connected. sockfd:=10
[INFO] Ethernet slave communication: start
[DEBUG] TM_COM: rv:=0
[DEBUG] TM_COM: Connection is ok
[DEBUG] TM_COM(Ethernet slave communication): 0_NONBLOCK connection is ok
[INFO] TM_COM(Ethernet slave communication): TM robot is connected. sockfd:=11

```

Step2

Terminal 2 :  
輸入  
source devel/setup.bash  
rosrn tm\_driver tm\_driver <ip>

## ROS1 連線(Melodic)

```

user@tclinux: ~/ros1
File Edit View Search Terminal Help
user@tclinux:~/ros1$ source devel/setup.bash
user@tclinux:~/ros1$ rosrn ui_for_debug_and_demo robot_ui

```

The QuickView\_GUI window shows the following status:

Connection Status		Re-Connect
Ethernet	On	TEXT
Listen Node	On	TEXT

Response ROS Node Status		
TMSVR	CPERR	DataErr
False	False	False
TMSCT/TMSTA	False	False

TM_Robot_Status Control_Box CQ Monitor		
MA_Mode	Project_Run	Project_Pause
MAN	On	Off
Robot_Link	Data_Table_Correct	ESTOP
True	True	Off
Robot_Error	Error_Code	Safeguard_A
False	False	Off

Step3

Terminal 3:

輸入

source devel/setup.bash

rosrun ui\_for\_debug\_and\_demo robot\_ui

此UI可以幫助使用者判斷連線是否成功

## ROS2 連線(Dashing)

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
user@tclinux:~/ros2$ source install/setup.bash
ROS_DISTRO was set to 'melodic' before. Please make sure that the environment does not mix paths from different distributions.
user@tclinux:~/ros2$ ros2 run tm_driver tm_driver 192.168.1.100
[INFO] [rclcpp]: TmCommunication::TmCommunication
[INFO] [rclcpp]: TmSBuffer::TmSBuffer
[INFO] [rclcpp]: TmCommRecv::TmCommRecv
[INFO] [rclcpp]: TmRobotState::TmRobotState
[INFO] [rclcpp]: Create DataTable
[INFO] [rclcpp]: Ethernet slave communication: TmSvrCommunication
[INFO] [rclcpp]: TmCommunication::TmCommunication
[INFO] [rclcpp]: TmSBuffer::TmSBuffer
[INFO] [rclcpp]: TmCommRecv::TmCommRecv
[INFO] [rclcpp]: Listen node communication: TmSctCommunication
[INFO] [rclcpp]: Ethernet slave communication: start
[INFO ONCE]TM_COM: ip:=192.168.1.100
[INFO] [rclcpp]: TM_COM: rv:=0
[INFO] [rclcpp]: TM_COM: Connection is ok
[INFO] [rclcpp]: TM_COM(Ethernet slave communication): O_NONBLOCK connection is ok
[INFO] [rclcpp]: TM_COM(Ethernet slave communication): TM robot is connected. so ckfd:=15
[INFO] [rclcpp]: TM_ROS: get data thread begin
[INFO] [rclcpp]: Listen node communication: start

```

Step1

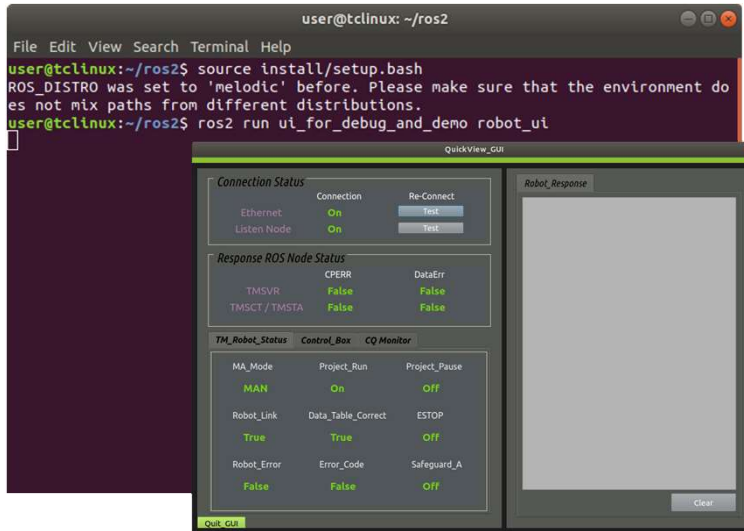
Terminal 1 :

輸入

source install/setup.bash

ros2 run tm\_driver tm\_driver <ip>

# ROS2 連線(Dashing)



Step2

Terminal 2:

輸入

source install/setup.bash

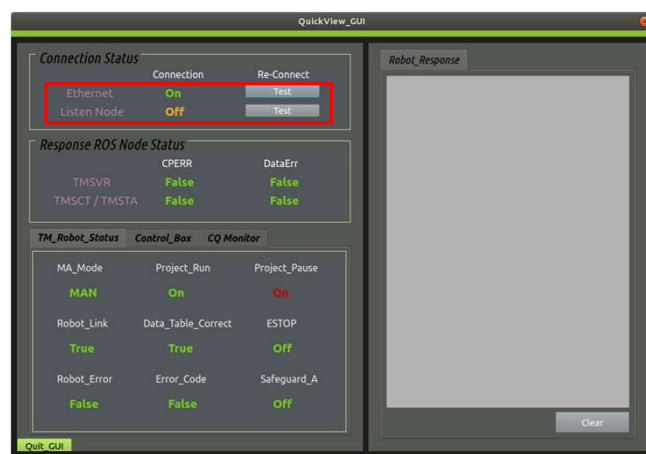
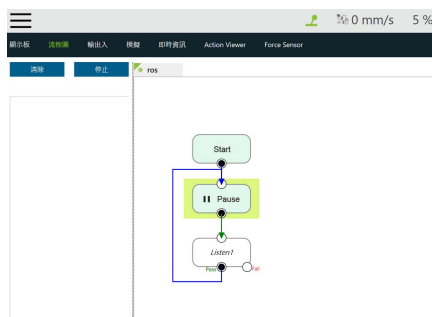
ros2 run ui\_for\_debug\_and\_demo robot\_ui

此UI可以幫助使用者判斷連線是否成功

# 連線常見問題

只要執行專案

就算沒有進入 listen node還是可以連線



## 連線常見問題

```
[INFO] [1659423600.22785298] [rclcpp]: TM_ROS: on listen node.
[INFO] [1659423601.755668351] [rclcpp]: TM_ROS: (Listen node): Reconnecting...
[INFO] [1659423606.277870817] [rclcpp]: TM_ROS: (Ethernet slave): Reconnecting...
[WARN] [1659423606.759555645] [rclcpp]: TM_COM: Connection timeout count=1
[INFO] [1659423606.759641643] [rclcpp]: TM_ROS: (Listen node): Reconnecting...
[WARN] [1659423611.303555642] [rclcpp]: TM_COM: Connection timeout count=1
[INFO] [1659423611.303644293] [rclcpp]: TM_ROS: (Ethernet slave): Reconnecting...
[WARN] [1659423611.783198477] [rclcpp]: TM_COM: Connection timeout count=2
[INFO] [1659423611.783294409] [rclcpp]: TM_ROS: (Listen node): Reconnecting...
```

連線失敗時

Listen node Reconnect  
Ethernet slave Reconnect

這兩個會一直跳

## 連線常見問題

當連線失敗要怎麼辦？

1. Ping IP 是否有通
2. Mask和IP是否一致
3. Ethernet slave是否設定完成並打開
4. 專案是否執行中
5. 換網路孔

# 04.

## ROS Vision

- ROS Vision Dependencies
- ROS Vision 使用說明

## ROS Vision Dependencies

- ROS2 Dashing (這邊以此版本進行示範)
- Python packages :
  - flask
  - waitress
  - opencv-python>=3.4.13.47(minimum)
  - numpy
  - datetime
- For example :
  - pip3 install flask
  - pip3 install waitress
  - pip3 install opencv-python
  - pip3 install datetime

## ROS Vision 使用說明

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
user@tclinux:~/ros2$ source install/setup.bash
ROS_DISTRO was set to 'melodic' before. Please make sure that the environment do
es not mix paths from different distributions.
user@tclinux:~/ros2$ ros2 run tm_get_status image_talker
Listening on an ip port:6189 combination
  
```

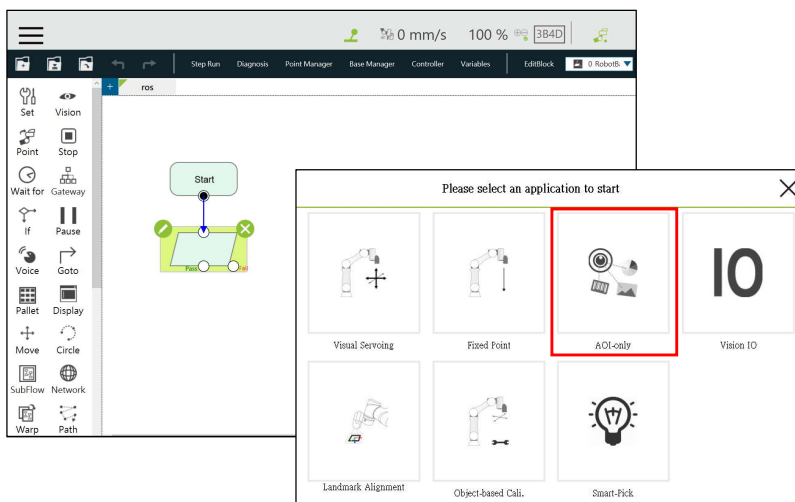
Step1

在Terminal輸入：

```

source install/setup.bash
ros2 run tm_get_status image_talker
  
```

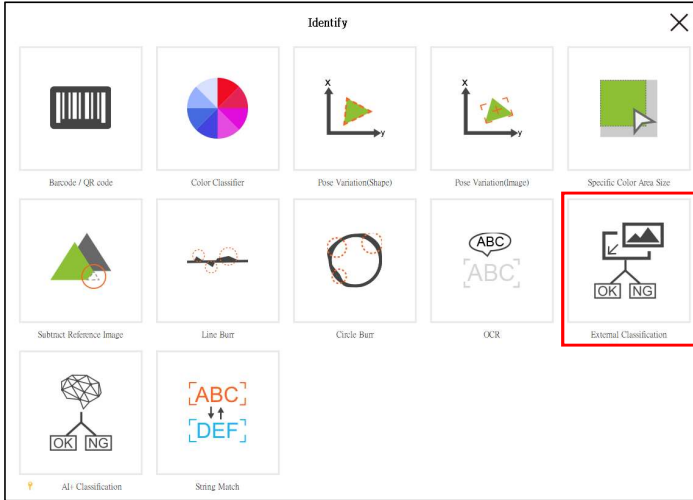
## ROS Vision 使用說明



Step2

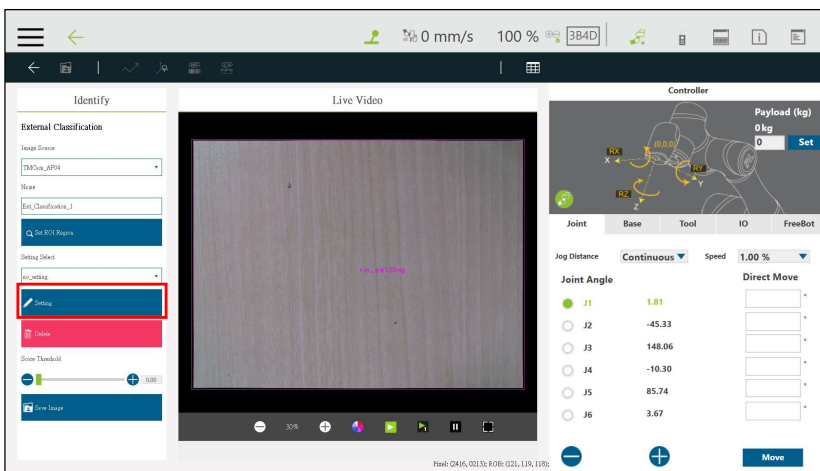
在TMFlow新增一個視覺任務，  
並選擇僅AOI辨識

# ROS Vision 使用說明



Step3  
選擇辨識模組·並選擇External Classification

# ROS Vision 使用說明



Step4  
點選Setting



## ROS Vision 使用說明

**HTTP Setting**

Get:

2022-10-13-10-35-52\_827 receive:  
JSON:  
{  
 "message": "running",  
 "result": "api"  
}

**Inference POST**

URL:

Post	Key	model_id	Value
		<input type="text"/>	<input type="text"/>

ipx  pax

Timeout(ms):

Setting name:

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
user@tclinux:~/ros2$ source install/setup.bash
ROS_DISTRO was set to 'melodic' before. Please make sure that the environment do
es not mix paths from different distributions.
user@tclinux:~/ros2$ ros2 run tm_get_status image_talker
Listening on an ip port:6189 combination
[192.168.1.100] [2022-10-13 10:40:47.760043] -> Get()
  
```

Step5

在Get中輸入 [http://\[IP\]:6189/api](http://[IP]:6189/api) ·  
可以測試連線是否成功

## ROS Vision 使用說明

**HTTP Setting**

Get:

2022-10-13-10-49-01\_578 receive:  
JSON:  
{  
 "message": "running",  
 "result": "api"  
}

**Inference POST**

URL:

Post	Key	model_id	Value
		<input type="text" value="test"/>	<input type="text" value="test"/>

ipx  pax

Timeout(ms):

Setting name:

Step6

確認連線沒有問題後 · 在下面的  
URL輸入[http://\[IP\]:6189/api/CLS](http://[IP]:6189/api/CLS) ·  
在Value中輸入test並按下Add ·  
最後再輸入一個setting name ·  
即可儲存本設定

## ROS Vision 使用說明

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
user@tclinux:~/ros2$ source install/setup.bash
ROS_DISTRO was set to 'melodic' before. Please make sure that the environment do
es not mix paths from different distributions.
user@tclinux:~/ros2$ ros2 run custom_package sub_img
width : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944
after set this->Image = frameWidth : 2592
Height: 1944

```



### Step7

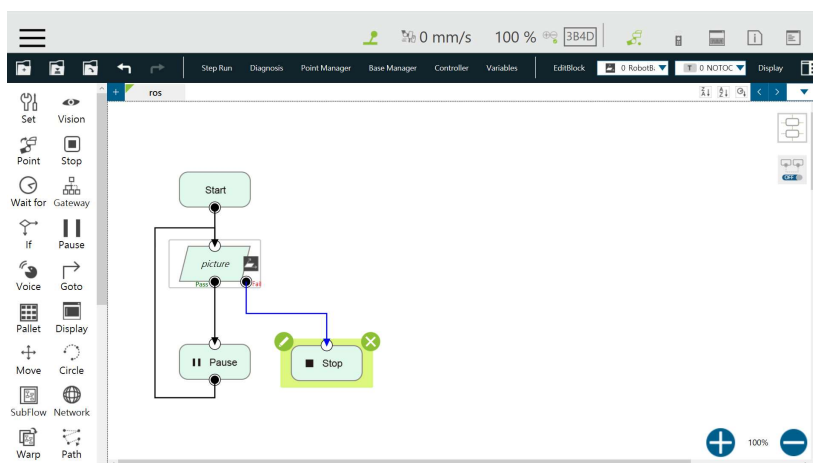
設定完成後，  
在Terminal輸入：

`source install/setup.bash`

`ros2 run custom_package sub_img`

可以讀取到傳送出來的圖片

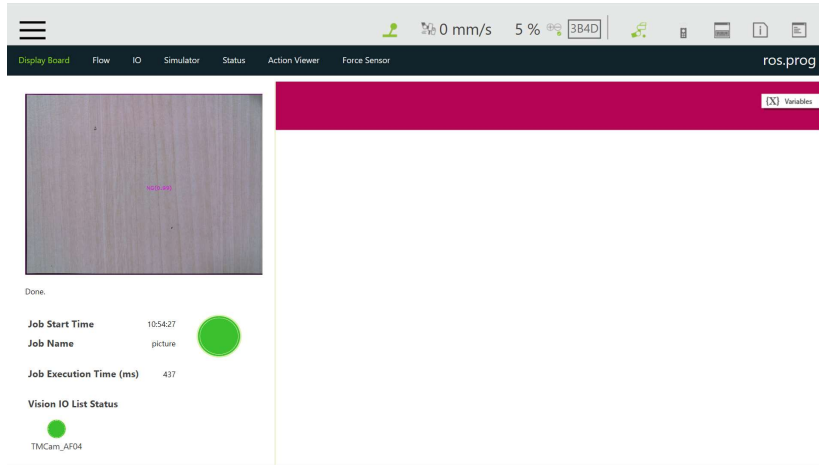
## ROS Vision 使用說明



### Step8

在專案中設定  
每按一下Play/Pause鍵取一張圖

# ROS Vision 使用説明



Step9  
執行專案

# 05. Moveit

- Moveit介紹
- Moveit使用説明



## Moveit介紹

目前moveit的支援程度:

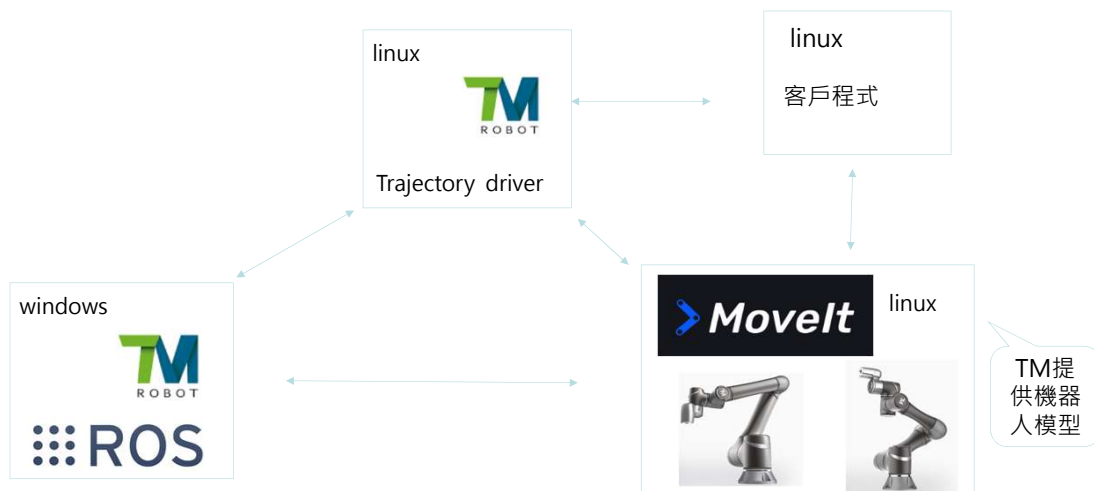
ROS2 : foxy (簡易顯示)和簡易demo

ROS2本身  
Moveit還沒完善

ROS1: melodic noetic(完整)

ROS1  
Moveit系統完整

## Moveit介紹



## Moveit使用說明

---

請參考Moveit的官方網站  
並安裝好所有需要用到的Package



[http://docs.ros.org/en/melodic/api/moveit\\_tutorials/html/doc/getting\\_started/getting\\_started.html](http://docs.ros.org/en/melodic/api/moveit_tutorials/html/doc/getting_started/getting_started.html)

## Moveit使用說明

---

這邊以Melodic進行示範

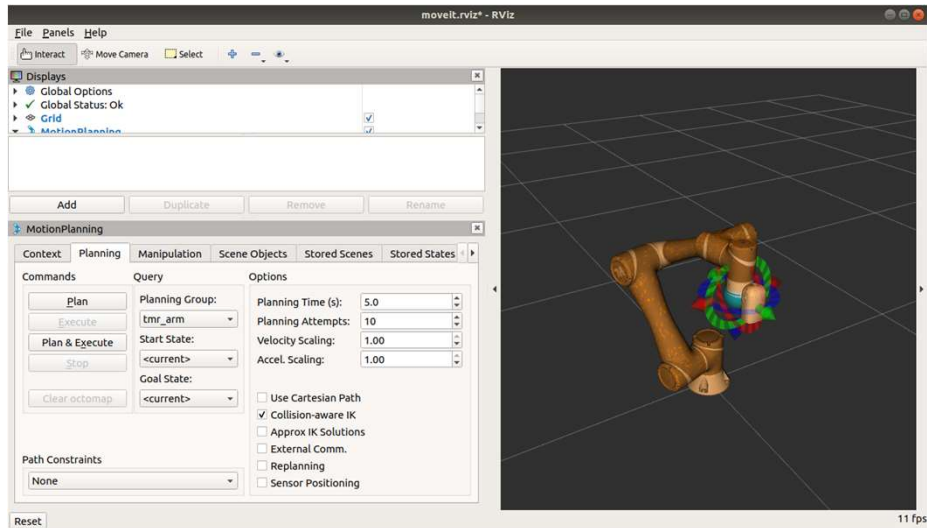
啟動虛擬的手臂

```
roslaunch tm5-900-moveit_config tm5-900_moveit_planning_execution.launch sim:= True
```

啟動真實的手臂

```
roslaunch tm5-900-moveit_config tm5-900_moveit_planning_execution.launch sim:=False robot_ip:= <robot_ip_address>
```

# Moveit使用説明



# 06.

## Demo code 使用

# Demo code 使用

```

Open  〇  *demo_send_script.cpp
//ros2demo

write (rcpp::wait_for_service{is}) {
  if (!rcpp::ok()) {
    RCLCPP_ERROR(rcpp::get_logger("rcpp"), "Interrupted while waiting for the service. Exiting.");
    return false;
  }
  RCLCPP_INFO(rcpp::get_logger("rcpp"), "service not available, waiting again...");
}

auto result = client->send_request(request);
// Wait for the result.
if (rcpp::spin_until_future_complete(node, result) ==
    rcpp::executor::FutureReturnCode::SUCCESS)
{
  //RCLCPP_INFO(rcpp::get_logger("rcpp"), "Sun: %d", result.get()->ok);
  if (result.get()->ok) {
    RCLCPP_INFO(rcpp::get_logger("rcpp"), "OK");
  } else {
    RCLCPP_INFO(rcpp::get_logger("rcpp"), "not OK");
  }
} else {
  RCLCPP_ERROR(rcpp::get_logger("rcpp"), "Failed to call service");
}
return true;
}

int main(int argc, char **argv)
{
  rclcpp::init(argc, argv);

  std::shared_ptr<rclcpp::Node> node = rclcpp::Node::make_shared("demo_send_script");
  rclcpp::client<tn_msgs::srv::SendScript>::SharedPtr client =
  node->create_client<tn_msgs::srv::SendScript>("send_script");
  std::string cmd = "TFP(\"JPP\",0,0,90,0,90,0,35,200,0,false)";
  send cmd(client, node, client);
}
    
```

這邊使用Ros2作為範例  
編輯src裡面的demo\_send\_script

# Demo code 使用

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
user@tclinux:~/ros2$ colcon build

user@tclinux: ~/ros2
File Edit View Search Terminal Help
/home/user/.local/lib/python3.6/site-packages/setuptools/dist.py:726: UserWarning:
Usage of dash-separated 'install-scripts' will not be supported in future versions.
Please use the underscore name 'install_scripts' instead
  % (opt, underscore_opt)
/home/user/.local/lib/python3.6/site-packages/setuptools/command/install.py:37:
SetuptoolsDeprecationWarning: setup.py install is deprecated. Use build and pip
and other standards-based tools.
  setuptools.SetuptoolsDeprecationWarning,
  ---
Finished <<< tn_get_status [1.44s]
Finished <<< technman_robot_msgs [1.73s]
Starting >>> custom_package
Finished <<< tn_msgs [2.18s]
Starting >>> demo
Starting >>> tn_driver
Starting >>> ut_for_debug_and_demo
Finished <<< custom_package [0.56s]
Finished <<< tn_driver [0.94s]
Finished <<< ut_for_debug_and_demo [0.98s]
Finished <<< demo [1.20s]

Summary: 7 packages finished [3.93s]
1 package had stderr output: tn_get_status
user@tclinux:~/ros2$
    
```

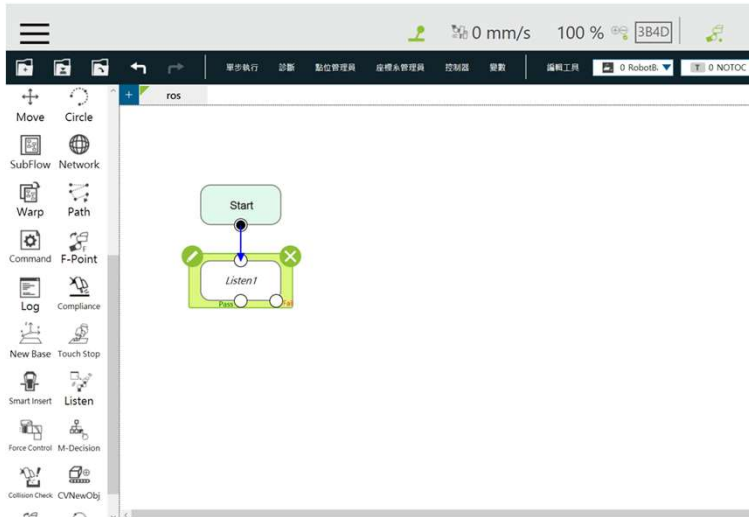
Step 1

編輯完專案後在Terminal輸入：

colcon build  
進行建置

建置完成

## Demo code 使用



Step 2

到TMFlow拉出一個Listen node  
並執行專案

## Demo code 使用

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
user@tclinux:~/ros2$ source install/setup.bash
ROS_DISTRO was set to 'melodic' before. Please make sure that the environment do
es not mix paths from different distributions.
user@tclinux:~/ros2$ ros2 run tm_driver tm_driver 192.168.1.100
[INFO] [rclcpp]: TmCommunication::TmCommunication
[INFO] [rclcpp]: TmSBuffer::TmSBuffer
[INFO] [rclcpp]: TmCommRecv::TmCommRecv
[INFO] [rclcpp]: TmRobotState::TmRobotState
[INFO] [rclcpp]: Create DataTable
[INFO] [rclcpp]: Ethernet slave communication: TmSvrCommunication
[INFO] [rclcpp]: TmCommunication::TmCommunication
[INFO] [rclcpp]: TmSBuffer::TmSBuffer
[INFO] [rclcpp]: TmCommRecv::TmCommRecv
[INFO] [rclcpp]: Listen node communication: TmSctCommunication
[INFO] [rclcpp]: Ethernet slave communication: start
[INFO_ONCE]TM_COM: ip:=192.168.1.100
[INFO] [rclcpp]: TM_COM: rv:=0
[INFO] [rclcpp]: TM_COM: Connection is ok
[INFO] [rclcpp]: TM_COM(Ethernet slave communication): 0_NONBLOCK connection is
ok
[INFO] [rclcpp]: TM_COM(Ethernet slave communication): TM robot is connected. so
ckfd:=15
[INFO] [rclcpp]: TM_ROS: get data thread begin
[INFO] [rclcpp]: Listen node communication: start

```

Step3

使用TM ROS driver進行連線

Terminal 1:  
輸入

source install/setup.bash  
ros2 run tm\_driver tm\_driver <ip>



## Demo code 使用

```

user@tclinux: ~/ros2
File Edit View Search Terminal Help
user@tclinux:~/ros2$ source install/setup.bash
ROS_DISTRO was set to 'melodic' before. Please make sure that the environment does not mix paths from different distributions.
user@tclinux:~/ros2$ ros2 run demo demo_send_script
[INFO] [rclcpp]: OK
user@tclinux:~/ros2$

```

Step4

執行demo\_send\_script

Terminal 2 :

輸入

source install/setup.bash

ros2 run demo demo\_send\_script

TECHMAN  
ROBOT



THANK YOU

TECHMAN  
ROBOT



[www.tm-robot.com](http://www.tm-robot.com)