# :robot: Image Processing: ROS2 Package

Project created for studies in the ROS2 platform. The project intends to crop images. The chosen crop method was a proportion crop starting from the center of the picture. The idea behind this is to concentrate focus on the part of the image with more information, and not just crop arbitrarily. Full disclosure, the choice of the center of the image was arbitrary, and based on the programmer's idea that the area around the center of the picture most of the time will hold more information.

Because the project proposal is a fully integrated project to measure the knowledge about the ROS2 platform and the concepts of Package, Nodes, Services, Topics, etc some best practices were not followed, like:

- Separated package to create custom srv and msg. Normally in a real ROS2 project when you are creating a custom srv or msg, the best way is to create a separate package to hold all custom srv and msg, this is because normally custom srv or msg can be used in different packages.
- Test implementation. Normally is a good practice to create tests for the packages.

Let's start understanding what cropping means. Cropping is the process of removing outer parts of the image to improve framing, alter the aspect ratio, or emphasize a particular subject. Below we have a few use cases for cropping:

- Focusing on key subject
- Composition improvements
- Data augmentation for deep learning
- Privacy protection

## ROS2 Installation

### Set the locale

```
locale  # check for UTF-8

sudo apt update && sudo apt install locales
sudo locale-gen en_US en_US.UTF-8
sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
export LANG=en_US.UTF-8

locale  # verify settings
```

## Setup Source

- Ensure that the Ubuntu Universe repository is enabled.

```
sudo apt install software-properties-common
sudo add-apt-repository universe
```

- Add ROS2 GPG key

```
sudo apt update && sudo apt install curl -y
sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key
-o /usr/share/keyrings/ros-archive-keyring.gpg
```

- Add repository to source list.

```
echo "deb [arch=$(dpkg --print-architecture)
signed-by=/usr/share/keyrings/ros-archive-keyring.gpg]
http://packages.ros.org/ros2/ubuntu $(. /etc/os-release && echo $UBUNTU_CODENAME)
main" | sudo tee /etc/apt/sources.list.d/ros2.list > /dev/null
```

## Install ROS2 packages

```
sudo apt update
sudo apt upgrade
sudo apt install ros-humble-desktop
sudo apt install ros-humble-ros-base
sudo apt install ros-dev-tools
```

## Setup Environment

- Run this snippet on every console you have open to have ros2 working on all consoles

```
# Replace ".bash" with your shell if you're not using bash
# Possible values are: setup.bash, setup.sh, setup.zsh
source /opt/ros/humble/setup.bash
```

- To have ros2 on every console without needing a source every time use gedit.

```
gedit ~/.bashrc
```

- In the .bashrc file add the following lines at the bottom.

```
#ROS2 - Humble
source /opt/ros/humble/setup.bash
source /usr/share/colcon_argcomplete/hook/colcon-argcomplete.bash
source ~/ros2_ws/install/setup.bash
```

- With those lines ros2 will be sourced on every console at the startup, colcon autocomplete will be active and every built package will be ready to be used.

### Install colcon

Make sure colcon is installed.

```
sudo apt install python3-colcon-common-extensions
```

## Starting Workspace

- To start the project let's first create the workspace. This project uses the default configuration and creates the **ros2_ws** at the system's home.

```
# at home
mkdir -p ~/ros2_ws/src
cd ~/ros2_ws
```

After that:

- Extract the image_processing_pkg.zip inside the **'~/ros2_ws/src'**.
- Rename the folder to **'image_processing_pkg'**

## How to run?

To run the project you will need to first be in the ros2 workspace, for this project it was used the name **'ros2_ws'** at the system's home.

- First, we need to build the package

```
colcon build --packages-select image_processing_pkg
```

### Setup before run the package

Inside the **image_processing_pkg_launch.xml** or **image_processing_pkg_launch.py** you can change the variable:

- **image_name**: the name of the image you want to crop (The project already has 4 example images to be used)
- **iterations_num**: iterations number
- **cropped_image_path**: Path where the cropped image will be saved

The image_processing_pkg_launch.* files are located inside **'ros2_ws/src/image_processing_pkg/launch'**

To add new images to be cropped, just add the image at the **'ros2_ws/src/image_processing_pkg/res/images'** and change the parameter image_name to the image's name you just added. Because we are providing a launch file through the package it's necessary to build the package every time the launch files are changed.

- After the package is built and the setup is done you can run the one of following commands:

```
# From ros2_ws
cd ~/ros2_ws
ros2 launch image_processing_pkg image_processing_pkg_launch.xml
ros2 launch image_processing_pkg image_processing_pkg_launch.py
```

## Project Diagrams

The diagrams of the project can be found in the **/docs** folder.

## What next?

An evolutive to the project could be add the ability to detect shapes, objects, texts, etc. This can be achieved by using the OpenCV lib. One approach to this could be:

- Convert it to a grayscale image
- Apply thresholding on the image and then find out the contours.
- Run a loop in the range of contours and iterate through it.
- In this loop draw an outline of shapes (Using drawContours() ) and find out the center point of the shape.
- Classify the detected shape based on the number of contour points it has and put the detected shape name at the center point of the shape.

- After that crop the shapes

## Links / References

- https://docs.ros.org/en/humble/
- https://answers.opencv.org/questions/
- https://answers.ros.org/questions/
- https://www.geeksforgeeks.org/naming-convention-in-c/