**Project topic: Robotics**

**Project description:**

"Think of a robot that was not programmed to do anything. The robot has a GPU, cameras, depth sensors and a robotic arm.

Use story: Put the robot on a conveyor built for a period of time. Let it learn what you wanted it to do by itself. Then deploy it.

How can you achieve this goal?"

1. **General Analysis**

As described, this project requires that the robot obtains information and eventually new skills on its own, or in other words, without being programmed to do so. AI is a possible method for this task. As defined in the Collins Dictionary, artificial intelligence is “a type of computer technology concerned with making machines work in an intelligent way, similar to the way that the human mind works.” Having the robot learn something should be classified as an application of AI.

Python is a popular programming language that is often used for AI. Thanks to a variety of packages donated by worldwide programmers, it is one of the most dynamic and effective languages in the world. Packages like tensorflow or OpenCV provide easy access to deep learning and image processing, and those like numpy offer convenient, packed functions for data processing. Although Python has relatively low speed, especially compared to C, it has very nice flexibility and is applicable in a very wide range of cases. It also allows for conveniently imbedding other languages and functions. Therefore, Python is a suitable programming for this project.

Machine learning is an important part of AI. It is based on training the target with massive data. There are many cases of successful applications of machine learning in image processing and recognition(Yuxin[育心]). Since the robot is not provided directly with data, it must acquire the information by itself, which means that the cameras and depth sensors will be the only way of observation. Cameras are more important than depth sensors because they intake more information at a time, and are vital in identifying objects. Therefore, they should be the main method of data acquisition.

Image processing can be made very easy by using the OpenCV package, along with other supportive packages. The data within the images should be extracted and used in the proper way, so that the robot could possibly understand what it is expected to do. Convolution Neural Network(CNN) is an effective way of doing this. It can generate new images by preserving only the targeted features. Different features are selected and preserved by using different convolution kernels. For example, some may strengthen the edge, another may turn the image into an outline. The convolution kernel(Laplace operator) strengthens the outline and turns the image units into either pure black or pure white. As exact details are not necessary and might jeopardize the robot’s ability to learn, it is appropriate to use said convolution kernel to extract only the useful information(the outlines of the image for example) while discarding other obsolete image units.

However, deep learning is not suitable for the entire project, because using this method requires a lot of existing, marked data so that the robot could identify what it is expected to identify. For example, the system is provided with a lot of pictures of dogs and cats, and is expected to teach itself to distinguish a dog from a cat. This project is not limited to identifying something, and clearly will not have prepared data. Although processing the images could be aided by CNN(commonly considered a part of deep learning), deep learning cannot solve the problem itself. The robot must find out on its own: what is the product; what are the other robots; what is the obsolete environment(or noise).

Deep learning is often considered as a part of machine learning although it has experienced great extension, and is considered by some to be an independent section in AI, rather than a subordinate part of machine learning(Benson-hdx).

Another part of machine learning is the reinforcement learning. Reinforcement learning is learning what to do—how to map situations to actions—so as to maximize a numerical reward signal(Richard S. Sutton and Andrew G. Barto). This method is applicable in the case when the system is expected to output an sequence of pre-set actions to achieve the best result. This is often used to find the best decision according to the specific environment. In this case, the best result refers to correctly processed products.

Supervised learning and unsupervised learning are also subordinate parts of machine learning. Supervised learning is used in cases where a “correct answer” is provided. The system is required to find a correct solution based on the given data that are all labeled.

Unsupervised learning gives data without labels to the system, and requires the system to find on itself some features. For example, a set of coordinates are given to the system, which needs to find a function that best describes their distribution.

1. **Mechanism, Principles and Stages**

The first stage is the learning stage, without any practice. The robot receives information via the cameras and depth sensors. After collecting an ample amount of information, the robot analyzes the information, and identifies the robot, the unprocessed products, the processed products, the obsolete environment(noise), etc. Then, the robot can understand what should be done to the products by comparing the images of the unprocessed ones and the processed ones. It can also learn how to perform by studying the images of other robots, if their motions are simple and periodic.

The second stage is the calibration of the robotic arm. There should be cameras and sensors surrounding the robotic arm to observe the performance of it. When an instruction is given to the arm, the robot should be given feedback of how fast and how far the arm moves. This step is the fundamental of accurate product processing.

For any project that requires accuracy, a close loop system is indispensable. The third stage is attempting to finish the task and calibrating the movement. By this stage, the robot should have learned what the task is and how to move the mechanical arm. Now it is time to put them into practice. In this stage, the robot corrects its movement, until it produces qualified products, which shall look the same as the processed products in stage one.

**2.1 Image Processing**

Identifying the different objects could be challenging. A traditional, less flexible way of doing this is to manually provide an algorithm for the robot to carry out. This could be reliable in some cases, but is likely to produce less satisfactory results, and will surely require further manual improvement in the code after testing. On the other hand, machine learning can be applied for adaptability and accuracy. Although no data is provided directly to the storage, the robot can acquire data with the cameras. However, these images will of course contain no mark or any further information, since the robot does not understand it yet. This makes it impossible for supervised learning, but unsupervised learning may be applied. Unsupervised learning is suitable for systems that seek features from completely unknown data. Since products could be various in shape, size and color(difference in color can be eliminated by transforming the image into a grayscale image), applying unsupervised learning promises relatively accurate results, whereas the traditional, manual way may well fail to apply to this situation without being further calibrated.

The first step for image processing and recognizing is to extract features, or in other words, identifying certain objects in an image. HOG features(usually aided with SVM), LBP features and Haar-like features are three types of image features that may be helpful in recognizing objects.

**2.1.1 HOG Features**

The HOG + SVM(Histogram of Oriented Features and Linear Support Vector Machine) method finds the features by calculating the gradients and forming a corresponding histogram. This method is particularly effective and widely applied in pedestrian detection. It is firstly discovered in 2005 by Navneet Dalal and Bill Triggers. Since products moving on the conveyor follows a similar but simpler pattern(without local movements or shape changes, e.g. moving arms and legs of pedestrians), it is promising to apply this method in this project. This function is integrated in the OpenCV package.

**2.1.2 LBP Features**

The LBP(Local Binary Pattern) method is used to describe local patterns of an image.This method was firstly discovered in 1994 by T. Ojala, M.Pietikäinen and D. Harwood. This is achieved initially by comparing the grayscale values of adjacent pixels in 3x3 matrixes. If a pixel is darker than the one in the middle, mark it with 1; otherwise mark it with 0. By repeating this procedure, an LBP atlas indicating each pixel’s LBP value can be acquired. This atlas (image) contains information of the original image’s features. Later, the 3x3 operator was improved and allowed selecting an arbitrary number of pixels in a zone within an arbitrary length from the targeted pixel. This makes the method immune to grayscale changes and rotation.

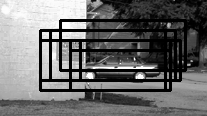
However, the LBP atlas is often not directly used to describe an image, because it is not immune to displacement. If displacement exists between two images that are in fact similar, they will not be considered similar because the positions of individual LBP values cannot match. LBP histograms are instead often used to describe an image. The LBP atlas is divided into several blocks, each described by a histogram that indicates numbers of respective LBP values. An image can be best described by these histograms.

**2.1.3 Haar-like Features**

Haar-like features indicate the grayscale changes of an image. This method best serves body part detection. Since in this project the robot needs to identify individual objects(products) as a whole and do not need to distinguish their parts, it is pointless to apply this method into this alien environment.

In conclusion, the HOG + SVM method is relatively mature and more reliable. Applying this method in this project is the best option.

After successfully identifying the products, another function called findContours in package OpenCV can be used to cut the contours of the product. In this way, irrelevant information(the background for example) could be clearly erased from the image. Another advantage of cutting along the contours is that sometimes the system may find multiple zones containing the same object, and that the object in these zones might have various locations, as shown in the following picture(Github username:bikz05).



<https://github.com/bikz05/object-detector>

If the part of the product in this image is cut off, uniqueness can be ensured. This can further ensure accuracy in the next stages by preventing the influence of the object’s location in an image.

**2.2 Correction and Calibration for Accuracy**

Machine learning can be applied for accuracy and automation. This can be done in the global level. For example, after the calibration of the mechanic arm, the robot is able to give precise instructions to it and make it move as expected. Because it is difficult to identify the product when it is being processed by other robots, directly finding out how the product is dealt with is impossible. Machine learning can be used to find out the correct procedures, because the “correct result”, which is the image of processed products, will have been acquired previously.

Apart from the general framework, calibration is also needed when it comes to the robotic arm. Given the driving program, the performance of the robotic arm should be monitored, in order to find out relations of parameters including time, power, angle, displacement, torque, etc. If this procedure was to be done with the help of AI, extra cameras, sensors and a lot of complicated program will be needed. It is much easier and more efficient for the people to do this manually.

Most mechanic arms are powered by motors that are directly driven and controlled by actuation modules, which contain an actuation chip as the core, L298N for an example. Actuation modules provide isolation between the motors and the command center of the robot. They receive orders from the command center, and gives the corresponding actuation power to the motors. PWM(Pulse Width Modulation) control is a good method for electrical control, because it can command the motor to output any torque or speed within the maximum capacity. Many single chip microcomputers provide integrated command lines for PWM output. For example, the STM32F103 series SCM generates PWM waves by linking a certain output pin to the integrated PWM module within. After configuring the I/O output options and integrated timer options, link the corresponding timer to the output pin, apply said command line, and the output pin will send out corresponding PWM waves.

In the driver program, close loop is also needed. The PID algorithm can be applied in the driver program of the robotic arm for accuracy, since motor motion and the arm motion do not necessarily have a linear relationship, and that the arm does not stop moving immediately when a stop command is given to the motor. Depth sensors can be used to detect the position of the mechanic arm. They can serve as the feedback of this system.

**2.3 Close Loop**

Close loop is essential for nearly any project. The robot must check if it is behaving properly through certain feedback. The feedback could be the similarity between the products it processed and those other robots processed.

Reinforcement learning best serves this stage. Reinforcement learning is a method that arranges behaviors to fit the environment for the best result. As mentioned, by this stage, the robot should have acquired sufficient data from the cameras, and have gained total control of the mechanical arm. The robot now understands what the “correct answer” is, because it now identifies the successfully processed products. The robot should then be able to arrange movements to pursue the “best solution”. The “best solution” is the set of movement that processes the products into ones that have 100% similarity(or an equivalent threshold) to those processed by other robots.

Supervised learning is not suitable for this stage.

**2.4 Feedback: Similarity Comparison**

The images of products processed by the robot should be compared with those of the products processed by other existing robots for similarity value. This can be done by perceptual hash algorithm. This algorithm is integrated in the OpenCV package.

Perceptual hash algorithm compares images by generating their respective “fingerprints”. The closer their fingerprints are, the more similar two images will be.The algorithm is realizes by the following steps:

1. Resize the image into 8\*8 pixels. This step erases any detail in the image and leaves but the structure and grayscale. This prevents influence of difference in sizes and proportions.
2. Simplify the colors. The 8\*8 image is turned into a grayscale image.
3. Calculate the average of 64 pixels.
4. Compare the grayscale of each pixel to said average. If the pixel is darker than or equal to the average, it is marked by 1. Otherwise it is marked by 0.
5. Combine the 64 bool values and form the fingerprint of the image.
6. Compare the fingerprints of two images for similarity. Similarity is higher when there are fewer differences in corresponding digits of the two fingerprints.

Having similarities quantified, it is possible to apply reinforcement learning in this project as a part of close loop.

1. **Summary**

The project requires three stages: image processing, calibration, and training. In the first stage, the robot takes in images using the cameras, analyses them using unsupervised learning, identifies different objects(processed products and unprocessed products), and labels these objects. The images of processed products are to be used in the final stage as reference. In the second stage, the mechanic arm should be calibrated. This stage can swop orders with the first one. In the final stage, the robot trains itself using reinforcement learning. The robot adopts this algorithm to find the most suitable method of processing the products. The most suitable method is defined as outputting the products that a nearly identical to those processed by other robots.

A self-learning robot is able to learn about its mission from existing robots. This technology can be effective when the current production line is to be expanded. Deploying self-learning robots allows the factory to multiply the production quickly without manually programming new robots. **This is especially advantageous when new robots are different from the existing ones**, since programming and testing new robots may be challenging. Successfully developing the technology of self-learning robots may provide great convenience, because, as mentioned, applying new robots that may well be different from the old ones will be made easier and will require little manual testing. Self-learning robots can also save the trouble of developing new programs when the production line needs to be changed for another purpose. For example, a factory owns several production lines with conveyors. It has several robots that are manually programmed, and other robots that can be self-learning. Should the factory decide to shop the current production and start manufacturing another product, all it needs to do is to find one single robot(or even humans! This project is result-oriented, meaning that the processing procedures are not important. The robot can figure it out itself) to do the new work, and let the self-learning robot learn this procedure. After this robot completes the learning, simply copy its strategies to other robots, and the new production line can be in operation quickly.

There are disadvantages as well. If programmed incorrectly, the robot might eventually fail to achieve what it is meant to do. Additionally, if the processing procedure is too complicated, it is possible that the robot cannot figure out what to do with the products.

In conclusion, self-learning robots working by conveyors may provide great convenience under the right circumstances, but will be applied in complicated environments with a lot of difficulty.

Works Cited

arag2009. *Three Algorithms for Image Feature Extraction: HOG Features, LBP Features and Haar Features[图像特征提取三大算法：HOG特征，LBP特征，Haar特征]*. CSDN Forum, 21 March 2017. web. 12 Sept. 2020

<https://blog.csdn.net/arag2009/article/details/64439221?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522160013570219724839851879%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=160013570219724839851879&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~first\_rank\_ecpm\_v3~rank\_business\_v1-3-64439221.ecpm\_v3\_rank\_business\_v1&utm\_term=%E5%9B%BE%E5%83%8F%E7%89%B9%E5%BE%81%E6%8F%90%E5%8F%96&spm=1018.2118.3001.4187>

ageitgey. *Image\_to\_numpy.* GitHub Forum, 10 Oct. 2019. web. 9 Sept. 2020

<https://github.com/ageitgey/image\_to\_numpy>

Benson-hbx. *Supervised Learning, Unsupervised Learning and Reinforcement Learning[监督学习、无监督学习和强化学习].* CSDN Forum, 26 Jan. 2018. web. 6 Sept. 2020

<https://blog.csdn.net/Baron\_hbx/article/details/79177385?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522160004796219195188323913%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request\_id=160004796219195188323913&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~first\_rank\_ecpm\_v3~rank\_business\_v1-2-79177385.ecpm\_v3\_rank\_business\_v1&utm\_term=%E7%9B%91%E7%9D%A3%E5%AD%A6%E4%B9%A0&spm=1018.2118.3001.4187>

bikz05. *Object-detector*. Github Forum, 23 Jul. 2015. web. 15 Sept. 2020

<https://github.com/bikz05/object-detector>

fengbingchun. *Perpetual Hash Algorithm Realization in OpenCV for Image Similarity Calculation[图像相似度计算之哈希值方法OpenCV实现]*. CSDN Forum, 25 December 2014. web. 16 Sept. 2020

<https://blog.csdn.net/fengbingchun/article/details/42153261>

Mjiansun. *Definition And Calling of Dynamic Variable Names in Python[Python动态变量名定义与调用].* CSDN Forum, 19 Sept. 2018. web. 13 Sept. 2020

<https://blog.csdn.net/u013066730/article/details/82776852?ops\_request\_misc=%257B%2522request%255Fid%2522%253A%2522159996209619725247408601%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request\_id=159996209619725247408601&biz\_id=0&utm\_medium=distribute.pc\_search\_result.none-task-blog-2~all~first\_rank\_ecpm\_v3~rank\_business\_v1-2-82776852.ecpm\_v3\_rank\_business\_v1&utm\_term=%E5%8A%A8%E6%80%81%E8%B0%83%E7%94%A8%E5%8F%98%E9%87%8F&spm=1018.2118.3001.4187>

Numpy. *Quickstart Tutorial*. Numpy Official Website, web. 12 Sept. 2020

<https://numpy.org/devdocs/user/quickstart.html>

OpenCV. *Introduction (Vers 4.5.0-pre).* OpenCV Official Website, web. 9 Sept. 2020

<https://docs.OpenCV.org/master/d1/dfb/intro.html>

OpenCV. *Image File Reading And Writing(Vers 4.5.0-pre).* OpenCV Official Website, web. 9 Sept.2020

<https://docs.OpenCV.org/master/d4/da8/group\_\_imgcodecs.html#ga61d9b0126a3e57d9277ac48327799c80>

Shuilanchengchengzhu[水蓝城城主]. *Summary of Common Convolution Kernels for Image Processing[常用图像卷积核小结].* CSDN Forum, 9 May. 2020. web. 8 Sept. 2020

<https://blog.csdn.net/mrliuzhao/article/details/106011640?utm\_medium=distribute.pc\_aggpage\_search\_result.none-task-blog-2~all~first\_rank\_v2~rank\_v25-4-106011640.nonecase&utm\_term=%E5%9B%BE%E5%83%8F%E5%A4%84%E7%90%86%E5%B8%B8%E7%94%A8%E5%8D%B7%E7%A7%AF%E6%A0%B8>

ttttthub. *Pedestrian-detection*. Github Forum, 24 Mar. 2019. web. 13 Sept. 2020

<https://github.com/ttttthub/pedestrian-detection>

Yuxin[育心]. *Differences Between AI, Machine Learning and Deep Learning[人工智能、机器学习和深度学习的区别].* Zhihu Forum, web. 6 Sept. 2020

<https://www.zhihu.com/tardis/landing/360/ans/249708509?query=%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0+%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0&mid=7ddd1483c175b221b721dc8da3769e06&guid=997462CADEF939809FBC0EFF7173A76F.1593615480273>