

**DESIGN OF A GRAPHICAL USER INTERFACE FOR COMPUTER NUMERICAL CONTROL (CNC) MACHINE**

**A PROJECT SUBMITTED TO DEPARTMENT OF ELECTRICAL ENGINEERING**

**WOLKITE UNIVERSITY**

In partial fulfilment of the requirements for 4th year semester projects of

Electrical and Computer Engineering

By

Tesfamichael Molla, Bereket Mekonnen, Dereje Gebreyes, Fikru Hamaro

April 2018

Wolkite University

College of Engineering and Technology

DESIGN OF A GRAPHICAL USER INTERFACE FOR COMPUTER NUMERICAL CONTROL (CNC) MACHINE

By

Tesfamichael Molla, Bereket Mekonnen, Dereje Gebreyes, Fikru Hamaro

Department of Electrical and Computer Engineering

Approved By:-

Mr. Beka / Advisor /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

A CNC machine is a “Computer Numerical Controlled” machine that is directed by software. A CNC in the old days were programmed through a language called G-code which is made up of a set of commands that move the CNC up, down, left and right. Machines are still run on G-code but current software's run it in the background making it much simpler to use a CNC machine. But, still those machine luck speed and accuracy despite the fact that they use a complex software. Our project proposes a new way running CNC machine with-out using any G-codes. Our system will have a very simple but efficient GUI (Graphical User Interface) built with python to control the machine, and an image processing using opencv python library.

# Introduction

Now-a-days we all can notice how far the manufacturing technology is growing, a lot of products are being available in the market within a short period time. It wouldn’t take a week for products to be released after their promotion. This is due to the increasingly growing high precision manufacturing machineries which are partial and fully automated. Time to time those machineries are being more accurate with respect to their speed and reliability.

Seeing the complexity of the design for materials that we use day-to-day we can easily imagine the increasing in highly automated high precision prototyping machines. Computer Numeric Controlled (CNC) Machines were the first major break-through in the field of rapid-prototyping. Before numerical control (in the case of punched tape technology) and computer numerical control (with analog and digital computing), parts had to be machined by hand. Many novel CNC specialty machines are built specifically for niche manufacturing processes. For example, electrochemical machining is used to cut highly durable metal products not otherwise feasible. Conventional CNC machines are more adept at and typically used for prototype development than manufacturing.

CNC Machines are electro-mechanical devices that manipulate machine shop tools using computer programming inputs. Almost every manufacturing industry uses CNC machines. With increase in the competitive environment and demands, the demand of CNC usage has increased to a greater extent. The machine tools that comes with the CNC are lathe, mills, shaper, welding etc. The industries that are using CNC machines are automotive industry, metal removing industries, industries of fabricating metals, [electrical discharge machining](http://www.mechanicalbooster.com/2017/04/electrical-discharge-machining.html) industries, wood industries etc.

CNC is the computerized control usually based on industrial computer. It could be special software and operation system inside dedicated to the process and the machine or Windows- / Linux- based (usually Windows) with the specific software and industrial hardware.

# 1.2 Problem statement

* Using G-code can increase the load on the microcontroller that would decrease the execution speed and also increase the cost due to the need for a high performance microcontroller.
* In case error occurred current CNC machine controllers mostly doesn’t save the current state, which forces them to start from the scratch after maintenance.
* In order to make CNC controllers compatible and flexible for a lot of machine types, they are built using a complex class structure, runs robust algorithms and uses a huge data arsenal.

# 1.3 Objectives

## 1.3.1 General objectives

The main objective of this project is to design and develop a graphical user interface to control a computer numerical control machine (CNC).

## 1.3.2 Specific objectives

Some of the specific objective of this project includes:

* Design a GUI using PyQt python library
* Implement an Edge detection on images
* Apply contour detection on the results from edge detection
* Convert the python files to .exe file

# 1.4 Significance of the project

It’s always been the greatest need in manufacturing industry, having a very fast, accurate and autonomous manufacturing machines and prototype designers. Our projects emphasis on building the machine that satisfies the huge need in the manufacturing industry. Regarding the past machines, the system works in a different way that will turn out to be a very accurate and fast decreasing the load that was on the microcontrollers.

# Project scope

Our project is built using an open source python modules which enables it to be more flexible than others. As an open source software its scope is unlimited, but as a project we are limiting the functionality just to manipulate the prototyping function. The system manages to save and load real time data to enable it manage sudden malfunctioning and start from where it’s left of. The project can be extended to run machine learning algorithms for different purposes and this be covered in recommendation.

# LITERATURE REVIEW

# 2.1 INTRODUCTION TO COMPUTER NUMERICAL CONTROL (CNC) MACHINES

## What Is NC?

* Numerical control (NC) refer to control of a machine or a process using symbolic codes consisting of characters and numerals.
* The concept of NC was proposed in the late 1940s by John Parsons who recommended a method of automatic machine control that would guide a milling cutter to produce a curvilinear motion in order to generate smooth profiles on the work-pieces.

## Components of Traditional NC System

* Part drawing (dimensions manufacturing notes)
* Written NC program for part manufacture
* Program punched onto tape
* Program instructions sent to NC machine

# 2.2 HISTORY OF COMPUTER NUMERICAL CONTROL (CNC) MACHINES

Numeric control is largely exploring work of a man, in the 1940s, persons conceived a method of using punched cards containing coordinate position data to control a machine tool. In 1948, parson demonstrated his concept to the US Air force, which subsequently sponsored a series of research project at servomechanism laboratory of the Massachusetts of technology (MIT).

The initial work at MIT involved the development of a prototype NC milling machine, by retrofitting a conventional tracer mill with position servo mechanisms for thee axes of the machine tool. The fist demonstration of the NC prototype was held in 1952. By 1953, the potential usefulness of the NC concept had been proven.

# 2.3 APPLICATION AREAS FOR THE FIRST NUMERIACL CONTROL (NC) MACHINE

* Commercial numerical control (NC) units
* Industries numerical control (NC) units
* Airframe builders
* Air force numerical control (NC) units

Air force continued its encouragement of NC machine. This work resulted in the automatically programmed tools (APT) language. The object of the APT research was to provide a means by which the part programmer could communicate the machining instruction to the machine tool in simple English statements. APT is still widely used in industries today. The most other modern part programming languages are also based on APT process.

## 2.4 CURRENT CNC MACHINE APPLICATION AREAS

* Robotics, welding machinery
* Grinders, EDM’s, flame cutters
* Inspection equipment and etc.…

# 2.5 CLASSIFICATION OF CNC MACHINES

* Classification based on the motion type.
* Classification based on the control loops.
* Classification based on the number of axes.
* Classification based on the power supply.

# Methodology

# Introduction

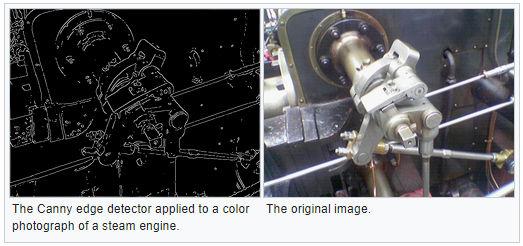
OpenCVOpenCV was started at Intel in 1999 by **Gary Bradsky** and the first release came out in 2000. **Vadim Pisarevsky** joined Gary Bradsky to manage Intel’s Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle who won 2005 DARPA Grand Challenge. Later its active development continued under the support of Willow Garage, with Gary Bradsky and Vadim Pisarevsky leading the project. Right now, OpenCV supports a lot of algorithms related to Computer Vision and Machine Learning and it is expanding day-by-day. Currently OpenCV supports a wide variety of programming languages like C++, Python, and Java etc. and is available on different platforms including Windows, Linux, OS X, Android, iOS etc. Also, interfaces based on CUDA and OpenCL are also under active development for high-speed GPU operations. OpenCV-Python is the Python API of OpenCV. It combines the best qualities of OpenCV C++ API and Python language.

OpenCV-Python (pytuto)Python is a general purpose programming language started by **Guido van Rossum**, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability. Compared to other languages like C/C++, Python is slower. But another important feature of Python is that it can be easily extended with C/C++. This feature helps us to write computationally intensive codes in C/C++ and create a Python wrapper for it so that we can use these wrappers as Python modules. This gives us two advantages: first, our code is as fast as original C/C++ code (since it is the actual C++ code working in background) and second, it is very easy to code in Python. This is how OpenCV-Python works, it is a Python wrapper around original C++ implementation. And the support of Numpy makes the task easier. **Numpy** is a highly optimized library for numerical operations. It gives a MATLAB-style syntax. All the OpenCV array structures are converted to-and-from Numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV, which increases number of weapons in your arsenal. Besides that, several other libraries like SciPy, Matplotlib which supports Numpy can be used with this. So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems.

Canny edge detection (https://en.wikipedia.org/wiki/Canny\_edge\_detector#Edge\_tracking\_by\_hysteresis)

The Canny edge detector is an [edge detection](https://en.wikipedia.org/wiki/Edge_detection) operator that uses a multi-stage [algorithm](https://en.wikipedia.org/wiki/Algorithm) to detect a wide range of edges in images. It was developed by [John F. Canny](https://en.wikipedia.org/wiki/John_F._Canny) in 1986. Canny also produced a *computational theory of edge detection* explaining why the technique works. Canny edge detection is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed. It has been widely applied in various computer vision systems. Canny has found that the requirements for the application of edge detection on diverse vision systems are relatively similar. Thus, an edge detection solution to address these requirements can be implemented in a wide range of situations. The general criteria for edge detection include:

1. Detection of edge with low error rate, which means that the detection should accurately catch as many edges shown in the image as possible
2. The edge point detected from the operator should accurately localize on the center of the edge.
3. A given edge in the image should only be marked once, and where possible, image noise should not create false edges.



To satisfy these requirements Canny used the [calculus of variations](https://en.wikipedia.org/wiki/Calculus_of_variations) – a technique which finds the [function](https://en.wikipedia.org/wiki/Function_(mathematics)) which optimizes a given [functional](https://en.wikipedia.org/wiki/Functional_(mathematics)). The optimal function in Canny's detector is described by the sum of four [exponential](https://en.wikipedia.org/wiki/Exponential_function) terms, but it can be approximated by the first [derivative](https://en.wikipedia.org/wiki/Derivative) of a [Gaussian](https://en.wikipedia.org/wiki/Gaussian_function).

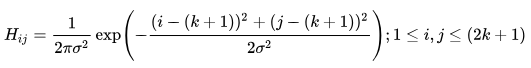
Among the edge detection methods developed so far, canny edge detection algorithm is one of the most strictly defined methods that provides good and reliable detection. Owing to its optimality to meet with the three criteria for edge detection and the simplicity of process for implementation, it became one of the most popular algorithms for edge detection.

# Process of Canny edge detection algorithm

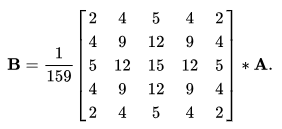
1. Apply Gaussian filter to smooth the image in order to remove the noise
2. Find the intensity gradients of the image
3. Apply non-maximum suppression to get rid of spurious response to edge detection
4. Apply double threshold to determine potential edges
5. Track edge by [hysteresis](https://en.wikipedia.org/wiki/Hysteresis): Finalize the detection of edges by suppressing all the other edges that are weak and not connected to strong edges.

# Gaussian filter

Since all edge detection results are easily affected by image noise, it is essential to filter out the noise to prevent false detection caused by noise. To smooth the image, a Gaussian filter is applied to convolve with the image. This step will slightly smooth the image to reduce the effects of obvious noise on the edge detector. The equation for a Gaussian filter kernel of size (2*k*+1)×(2*k*+1) is given by:



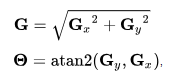
Here is an example of a 5×5 Gaussian filter, used to create the adjacent image, with {\displaystyle \sigma }ϭ = 1.4. (The asterisk denotes a [convolution](https://en.wikipedia.org/wiki/Convolution) operation.)



It is important to understand that the selection of the size of the Gaussian kernel will affect the performance of the detector. The larger the size is, the lower the detector’s sensitivity to noise. Additionally, the localization error to detect the edge will slightly increase with the increase of the Gaussian filter kernel size. A 5×5 is a good size for most cases, but this will also vary depending on specific situations.

## Finding the intensity gradient of the image

An edge in an image may point in a variety of directions, so the Canny algorithm uses four filters to detect horizontal, vertical and diagonal edges in the blurred image. The [edge detection operator](https://en.wikipedia.org/wiki/Edge_detection) (such as [Roberts](https://en.wikipedia.org/wiki/Roberts_Cross), [Prewitt](https://en.wikipedia.org/wiki/Prewitt), or [Sobel](https://en.wikipedia.org/wiki/Sobel_operator)) returns a value for the first derivative in the horizontal direction (G*x*) and the vertical direction (G*y*). From this the edge gradient and direction can be determined:



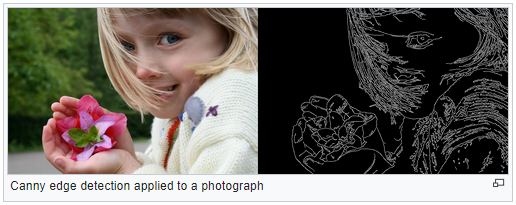
Where G can be computed using the [hypot](https://en.wikipedia.org/wiki/Hypot" \o "Hypot) function and [atan2](https://en.wikipedia.org/wiki/Atan2) is the arctangent function with two arguments. The edge direction angle is rounded to one of four angles representing vertical, horizontal and the two diagonals (0°, 45°, 90° and 135°). An edge direction falling in each color region will be set to a specific angle values, for instance θ in [0°, 22.5°] or [157.5°, 180°] maps to 0°.

# Double threshold

After application of non-maximum suppression, remaining edge pixels provide a more accurate representation of real edges in an image. However, some edge pixels remain that are caused by noise and color variation. In order to account for these spurious responses, it is essential to filter out edge pixels with a weak gradient value and preserve edge pixels with a high gradient value. This is accomplished by selecting high and low threshold values. If an edge pixel’s gradient value is higher than the high threshold value, it is marked as a strong edge pixel. If an edge pixel’s gradient value is smaller than the high threshold value and larger than the low threshold value, it is marked as a weak edge pixel. If an edge pixel's value is smaller than the low threshold value, it will be suppressed. The two threshold values are empirically determined and their definition will depend on the content of a given input image.

# Edge tracking by hysteresis

So far, the strong edge pixels should certainly be involved in the final edge image, as they are extracted from the true edges in the image. However, there will be some debate on the weak edge pixels, as these pixels can either be extracted from the true edge, or the noise/color variations. To achieve an accurate result, the weak edges caused by the latter reasons should be removed. Usually a weak edge pixel caused from true edges will be connected to a strong edge pixel while noise responses are unconnected. To track the edge connection, [blob analysis](https://en.wikipedia.org/wiki/Connected-component_labeling)is applied by looking at a weak edge pixel and its 8-connected neighborhood pixels. As long as there is one strong edge pixel that is involved in the blob, that weak edge point can be identified as one that should be preserved.



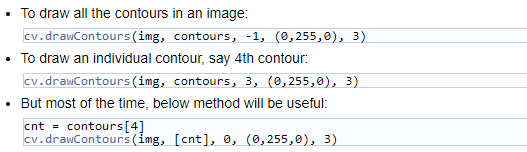
# What are contours? (<https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html> )

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition.

* For better accuracy, use binary images. So before finding contours, apply threshold or canny edge detection.
* Since OpenCV 3.2 [*findContours*](https://docs.opencv.org/3.4/d3/dc0/group__imgproc__shape.html#ga17ed9f5d79ae97bd4c7cf18403e1689a)no longer modifies the source image but returns a modified image as the first of three return parameters.
* In OpenCV, finding contours is like finding white object from black background. So remember, object to be found should be white and background should be black.

# How to draw the contours?

To draw the contours, [*cv.drawContours*](https://docs.opencv.org/3.4/d6/d6e/group__imgproc__draw.html#ga746c0625f1781f1ffc9056259103edbc)function is used. It can also be used to draw any shape provided you have its boundary points. Its first argument is source image, second argument is the contours which should be passed as a Python list, third argument is index of contours (useful when drawing individual contour. To draw all contours, pass -1) and remaining arguments are color, thickness etc.



# Work plan

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Months → | April | | | | May | | | |
| Weeks → | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Literature review |  |  |  |  |  |  |  |  |
| Edge detection |  |  |  |  |  |  |  |  |
| Contour detection and manipulation |  |  |  |  |  |  |  |  |
| GUI Modeling |  |  |  |  |  |  |  |  |
| GUI Implementation |  |  |  |  |  |  |  |  |