**MỤC LỤC**

[LỜI MỞ ĐẦU 1](#_Toc103167891)

[Chương 1:TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH LABVIEW 4](#_Toc103167892)

[1.1. Tổng quan về labVIEW 4](#_Toc103167893)

[1.1.1. LabVIEW là gì? 4](#_Toc103167894)

[1.1.2. Vai trò của labVIEW 4](#_Toc103167895)

[1.1.3. Các chức năng chính của LabVIEW 5](#_Toc103167896)

[1.1.4. Các công cụ liên kết với LabVIEW 6](#_Toc103167897)

[1.1.5. Các giao thức kết nối được sử dụng trong LabVIEW 6](#_Toc103167898)

[1.1.6. Các Module và các Toolkit hỗ trợ 7](#_Toc103167899)

[1.1.6.1. Các module chính 7](#_Toc103167900)

[1.1.6.2. Các Toolkit hỗ trợ người dùng 7](#_Toc103167901)

[1.1.7. Làm việc với LabVIEW 8](#_Toc103167902)

[1.2. Các thành phần chính của LabVIEW 9](#_Toc103167903)

[1.2.1. Bảng giao diện (The Front Panel) 9](#_Toc103167904)

[1.2.2. Sơ đồ khối (The Block Diagram) 11](#_Toc103167905)

[1.3. Những công cụ hỗ trợ lập trình labVIEW 13](#_Toc103167906)

[1.3.1 Tools Palette 13](#_Toc103167907)

[1.3.2. Bảng điều khiển (Controls Palette) 14](#_Toc103167908)

[1.3.3. Bảng các hàm chức năng (Function palette) 18](#_Toc103167909)

[1.4. Các kiểu dữ liệu 19](#_Toc103167910)

[1.4.1. Variables (biến) 19](#_Toc103167911)

[1.4.2. String 20](#_Toc103167912)

[1.4.3. Array 21](#_Toc103167913)

[1.4.4. Các cấu trúc điều khiển luồng chương trình 22](#_Toc103167914)

[1.5. SubVI và cách xây dựng SubVI 22](#_Toc103167915)

[1.5.1. Khái niệm SubVI 22](#_Toc103167916)

[1.5.2. Xây dựng SubVI 23](#_Toc103167917)

[Kết luận chương 1 25](#_Toc103167918)

[Chương 2: THỊ GIÁC MÁY TÍNH TRONG LABVIEW 27](#_Toc103167920)

[2.1. Tổng quan về xử xử lý ảnh 27](#_Toc103167921)

[2.1.1. Xử lý ảnh và quá trình phát triển 27](#_Toc103167922)

[2.1.2. Những vấn đề cơ bản trong hệ thống xử lý ảnh. 28](#_Toc103167923)

[2.1.3. Các giai đoạn xử lý dùng trong hệ thống xử lý ảnh 31](#_Toc103167924)

[2.2. Thị giác máy tính trong công nghiệp. 33](#_Toc103167925)

[2.2.1. Tầm quan trọng của thị giác máy tính 33](#_Toc103167926)

[2.2.2. Các ứng dụng chính của thị giác máy tính trong thực tiễn 35](#_Toc103167927)

[2.3. Các Module hỗ trợ thị giác máy tính trong LabVIEW 38](#_Toc103167928)

[2.3.1. NI Vision Development Module 38](#_Toc103167929)

[2.3.2. NI Vision Acquisition Software 43](#_Toc103167930)

[2.4. Một số kỹ thuật dùng trong thị giác máy tính cho bài toán chẩn đoán lỗi 45](#_Toc103167931)

[2.4.1. Phương pháp Edge Detection 46](#_Toc103167932)

[2.4.2. Phương pháp Pattern Matching 48](#_Toc103167933)

[2.4.3. Phương pháp Image subtraction 50](#_Toc103167934)

[Kết luận chương 2 51](#_Toc103167935)

[Chương 3: ỨNG DỤNG THỊ GIÁC MÁY TÍNH CHO BÀI TOÁN CHẨN ĐOÁN LỖI DÁN NHÃN SẢN PHẨM 52](#_Toc103167937)

[3.1. Tổng quan về hệ thống xây dựng 52](#_Toc103167938)

[3.1.1. Máy tính (computer) 52](#_Toc103167941)

[3.1.2. Camera (webcam) 54](#_Toc103167946)

[3.1.3. Thiết bị chiếu sáng 56](#_Toc103167947)

[3.2. Giải pháp chẩn đoán lỗi dán nhãn sản phẩm 56](#_Toc103167951)

[3.2.1. Đối tượng thực nghiệm 56](#_Toc103167952)

[3.2.2. Giải pháp chẩn đoán lỗi dán nhãn sản phẩm 58](#_Toc103167953)

[3.2.2.1. Khâu xây dựng kết nối với cổng Webcam 59](#_Toc103167954)

[3.2.2.2. Khâu đọc Barcode và truy xuất thư viện Barcode 61](#_Toc103167955)

[3.2.2.3. Khâu xử lý ảnh Vision Assistant chẩn đoán lỗi dán nhãn 62](#_Toc103167956)

[3.2.2.4 Xây dựng giải thuật cho bài toán chẩn đoán lỗi nhãn dán bị rách 63](#_Toc103167957)

[3.2.2.5. Xây dựng giải thuật cho bài toán chẩn đoán lỗi dán nhãn bị lệch 64](#_Toc103167958)

[3.3. Mô hình thực nghiệm 65](#_Toc103167959)

[3.3.1. Mô hình thực tế 65](#_Toc103167960)

[3.3.2. Giao diện người dùng trên Front Panel 66](#_Toc103167961)

[3.3.3. Một số kết quả thực nghiệm 68](#_Toc103167962)

[Kết luận chương 3 70](#_Toc103167963)

[KẾT LUẬN CHUNG 70](#_Toc103167966)

[TÀI LIỆU THAM KHẢO 72](#_Toc103167967)

**MỤC LỤC HÌNH ẢNH**

Hình 1.1: Phạm vi ứng dụng của LabVIEW 5

Hình 1.2: Bảng giao diện mới 10

Hình 1.3: Thanh công cụ giao diện 11

Hình 1.4: Sơ đồ khối của LabVIEW 12

Hình 1.5: Các dạng dây nối trên sơ đồ khối 13

Hình 1.6: Hình Tools Palette 13

Hình 1.7: Bảng mẫu Controls 15

Hình 1.8: Bảng điều khiển và chỉ thị số 16

Hình 1.9: Bảng điều khiển và chỉ thị logic 16

Hình 1.10: Graph 17

Hình 1.11:Bảng Function 18

Hình 1.12: giao diện SubVI 23

Hình 1.13: Icon subVI 24

Hình 1.14: một chương trình subVI 25

Hình 2.1: Ví dụ về điểm ảnh 29

Hình 2.2: Ví dụ về ảnh màu 30

Hình 2.3: Sơ đồ quá trình xử lý của hệ thống thị giác máy tính 31

Hình 2.4: Sơ đồ mô tả hệ thống ứng dụng thị giác máy trong sản xuất 36

Hình 2.5: Ứng dụng thị giác máy tính trong quá trình chẩn đoán lỗi dán nhãn 37

Hình 2.6: Ứng dụng trong tự động hóa sản xuất ô tô công nghệ 4.0 37

Hình 2.7: Giao diện bên trong NI Package Mangager 39

Hình 2.8: LabVIEW and Driver 39

Hình 2.9: Sủ dụng thanh tìm kiếm cho Vision 40

Hình 2.10: Vision and Motion 40

Hình 2.11: Các khối chức năng trong Vision Utilities 41

Hình 2.12: Các khối chức năng trong Image Processing 42

Hình 2.13: Các khối chức năng trong Machine Vision 42

Hình 2.14: Các khối chức năng trong NI-IMAQdx 43

Hình 2.15: Các khối chức năng trong Vision Express 43

Hình 2.16: Giao diện tìm kiếm NI Vision Acquisition Software 44

Hình 2.17: NI Vision Acquisition Software 44

Hình 2.18: Chương trình bên Block Diagram 46

Hình 2.19: Khối IMAQ ROIProfile 47

Hình 2.20: Khối IMAQ Edge Tool 3 47

Hình 2.21: Khối Overlay Points with User Specified Size 47

Hình 2.22: IMAQ Dispose 47

Hình 2.23: Giao diện chương trình bên Front Panel 48

Hình 2.24: Giao diện chương trình Pattern Matching bên Front Panel 49

Hình2.25: Giao diện chương trình Pattern Matching bên Block Diagram 49

Hình 2.26: Ảnh khi được Subtract 50

Hình 2.27: Hình ảnh mẫu 50

Hình 2.28: Hình ảnh có vết bẩn và hình đã được qua xử lý ảnh bằng phương pháp Suctract 51

Hình 3.1: Tổng quan về hệ thống xây dựng 52

Hình 3.2: Camputer 53

Hình 3.3: Laptop 53

Hình 3.4: Webcam Rapoo C270L 55

Hình 3.5: Một số nhãn tiêu biểu 57

Hình 3. 6: Hình ảnh sản phẩm lọ tương ớt Chin-Su 57

Hình 3.7: Một số hình ảnh về lọ tương ớt 58

Hình 3.8: Lưu đồ thuật toán 59

Hình 3.9: Thiết lập kết nối với Webcam 60

Hình 3.10: Đọc Barcode 61

Hình 3.11: Truy xuất thư viện 62

Hình 3.12: Khối xử lý ảnh Vision chẩn đoán lỗi dán nhãn 62

Hình 3.13: Thiết lập thuật toán chẩn đoán lỗi rách nhãn dán 63

Hình 3.14: Thiết lập thuật toán chẩn đoán lỗi lệch nhãn dán 65

Hình 3.15: Mô hình thực tế 66

Hình 3.16: Hệ thống ánh sáng 67

Hình 3.17: Giao diện người dùng trên Front Panel 68

Hình 3.18: Giao diện thông báo sản phẩm đạt chuẩn 69

Hình 3.19: Giao diện thông báo nhãn dán sản phẩm bị rách 69

Hình 3.20: Giao diện thông báo nhãn dán sản phẩm bị dán lệch 70

# LỜI MỞ ĐẦU

Hiện nay, nước ta đang trong quá trình công nghiệp hóa –hiện đại hóa đất nước, vai trò của lao động sáng tạo đang được đẩy mạnh. Việc cần thiết lúc này được đặt ra cho tất cả chúng ta là cải tiến trong công việc, nâng cấp công nghệ và hợp lý hóa sản xuất, nâng cao chất lượng , tăng năng suất lao động. Đây là một thách thức cũng như là một cơ hội cho ngành khoa học kỹ thuật ở nước ta. Ta có thể thấy được ngày nay càng nhiều công ty mới xuất hiện từ các quốc gia như Mỹ, Anh, Đức, Trung Quốc đang cạnh tranh về cả công nghệ và giá thành. Do vậy yêu cầu không ngừng cải tiến thiết bị máy móc tại các phân xưởng sản xuất nhằm nâng cao hiệu quả sản xuất và giảm chi phí lao động đang là vấn đề hết sức được chú trọng ngay lúc này.

Bên cạnh quá trình công nghiệp hóa – hiện đại hóa và thời kỳ công nghệ 4.0 hiện nay thì nhu cầu của khách hàng ngày càng khắc khe đối với các sản phẩm được bán trên thị trường, sự đòi hỏi đó bao gồm về chất lượng sản phẩm phải tốt, mẫu hình đẹp, dễ dàng sử dụng, và giá thành phải hợp lý. Vì vậy, để đáp ứng được như cầu đó thì các tiêu chuẩn trong kiểm tra và đánh giá chất lượng phải được chú trọng và đẩy mạnh hơn nữa. Trong các khâu sản xuất thì khâu kiểm tra và truy vết lỗi sản phẩm đòi hỏi được thực hiện một cách chỉnh chu và kỹ lưỡng trước khi giao sản phẩm đến khách hàng. Trong các nhà máy hiện nay, chủ yếu vẫn sử dụng hình thức thủ công để kiểm tra công đoạn dán nhãn cho sản phẩm trong giai đoạn đóng gói. Giai đoạn này đòi hỏi sự tập trung cao của công nhân để kiểm tra sự sai sót trong việc in ấn và dán nhãn, mà trong đó các lỗi hay gặp như: nhãn in mờ, không rõ nét, nhãn không in và sai Barcode sản phẩm. Do vậy các sản phẩm đóng gói không đạt yêu cầu có thể được gửi đến khách hàng và điều này ảnh hưởng rất nghiêm trọng đến uy tín của công ty, có thể dẫn đến các khoản đền bù lớn do những sai sót này.

Vì vậy, xuất phát nhu cầu thực tế trên, em đã chọn đề tài “**Nghiên cứu và ứng dụng thị giác máy tính để chẩn đoán lỗi dán nhãn sản phẩm**” để nghiên cứu và thực hiện đồ án tốt nghiệp.

**Nội dung nghiên cứu**

**- Mục tiêu nghiên cứu.**

* Thiết kế chương trình chẩn đoán lỗi dán nhãn sản phẩm với các lỗi: sai mã vạch (Barcode), in nhãn bị mờ, dán nhãn bị lệch, nhãn dán bị rách.

**- Đối tượng nghiên cứu**

* Nghiên cứu ứng dụng LabVIEW trong việc xử lý ảnh
* Nghiên cứu thị giác máy tính trong công nghiệp sản xuất

**- Phạm vị nghiên cứu**

* Nghiên cứu xử lý ảnh trên phần mềm LabVIEW để phân tích và kiểm tra, đo lường, đánh giá dán nhãn sản phẩm trong dây truyền sản xuất.
* Nghiên cứu tiến hành việc mô phỏng trên labVIEW và chạy thực nghiệm với một chiếc Webcam để thu hình ảnh và truyền hình ảnh vào chương trình labVIEW để xử lý hình ảnh.

**- Phương pháp nghiên cứu**

Kết hợp phương pháp lý thuyết và thực nghiệm:

* Phương pháp lý thuyết: Cơ sở lý thuyết ứng dụng LabVIEW trong phân tích xử lý hình ảnh để chẩn đoán lỗi dán nhãn sản phẩm. Đưa ra hướng giải quyết bài toán.
* Phương pháp thực nghiệm: Thử nghiệm với hình ảnh sản phẩm đã được chụp sẵn để chạy thử trên chương trình labVIEW. Sau đó, chạy thử nghiệm thực tế với Webcam để biết các mức độ nhận diện so sánh và đánh giá.

**- Ý nghĩa khoa học**

* Làm chủ công nghệ điều khiển thiết bị kiểm tra nhãn dán sản phẩm với hệ thống kiểm tra Vision.

**- Ý nghĩa thực tế**

* Hạn chế những sai sót nhầm lẫn ở công đoạn in nhãn, nâng cao chất lượng của quy trình kiểm soát ở công đoạn đóng gói thành phẩm. Nâng cao uy tín của công ty đối với khách hàng.
* Chi phí đầu tư thấp giảm đến 50% chi phí đầu tư cho 1 hệ thống robot scara có trang bị hệ thống vision.
* Giảm chi phí nhân công 1 người ở công đoạn kiểm tra nhãn in sản phẩm. Ví dụ: Toàn nhà máy có 30 dây chuyền, như vậy giảm được 30 người và chi phí ước tính cho một nhân công 5 triệu/tháng. Tiết kiệm 150 triệu/tháng.

**Nội dung thực hiện đề tài của em gồm 3 chương:**

Chương 1: Tổng quan về ngôn ngữ lập trình LabVIEW

Chương 2: Thị giác máy tính trong LabVIEW

Chương 3:Ứng dụng thị giác máy tính cho bài toán chẩn đoán lỗi dán nhãn sản phẩm

Do thời gian có hạn và kiến thức chuyên môn cũng như thực tế còn hạn chế nên bản đồ án này không tránh khỏi những thiếu sót. Kính mong nhận được sự chỉ bảo, góp ý của các thầy cô và các bạn để đồ án này được hoàn thiện hơn.

Em xin chân thành cảm ơn sự giúp đỡ chỉ bảo tận tình của các thầy cô, đặc biệt là thầy **Phạm Xuân Thủy** đã tạo điều khiện tốt nhất để em hoàn thành đồ án này. Em xin kính chúc các thầy cô luôn luôn mạnh khỏe để có thể tiếp tục dìu dắt nhiều thế hệ sinh viên.

# 

# Chương 1:TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH LABVIEW

## 1.1. Tổng quan về labVIEW

### 1.1.1. LabVIEW là gì?

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) là ngôn ngữ lập trình đồ họa sử dụng các biểu tượng (Icon) thay cho những dòng lệnh để tạo ứng dụng.

### 1.1.2. Vai trò của labVIEW

LabVIEW là một phần mềm được sử dụng rộng rãi trên thế giới vì nó mang lại nhiều lợi ích từ nghiên cứu trên hình thức mô phỏng cho đến chạy thực tế bên ngoài. Cho lên phần mềm LabVIEW được áp dụng cho các chương trình học tập và làm việc. Dưới đây là một trong những khả năng hỗ trợ cho các yêu cầu bài toán khác nhau:

* Kiểm tra, đo kiểm và phân tích tín hiệu trong kỹ thuật (đo nhiệt độ, phân tích nhiệt độ trong ngày).
* Thu thập dữ liệu (Data Acquisition), (thu thập các giá trị áp suất, cường độ, dòng điện,…).
* Điều khiển các thiết bị ( điều khiển động cơ DC, điều khiển nhiệt độ trong lò …).
* Phân loại sản phẩm (dùng chương trình xử lý ảnh để phân biệt sản phẩm bị lỗi, phế phẩm).
* Báo cáo trong công nghiệp (thu thập, phân tích xử lý dữ liệu và báo cáo cho người quản lý ở rất xa thông qua giao thức truyền TCP/IP trong môi trường mạng Ethernet).
* Giao tiếp máy tính và truyền dẫn dữ liệu qua các cổng giao tiếp (hỗ trợ hầu hết các chuẩn giao tiếp như USB, PCI, COM, RS-232, RS-485).



Hình 1.1: Phạm vi ứng dụng của LabVIEW

### 1.1.3. Các chức năng chính của LabVIEW

Dưới đây là những chức năng chính của phần mềm LabVIEW có thể đem lại. Tuy vậy, không chỉ có những chức năng chính này mà còn rất nhiều chức năng khác nữa chưa được tìm kiếm sâu và phần mềm đang được phát triển trong các phiên bản sau này:

* Thu thập tín hiệu từ các thiết bị bên ngoài như cảm biến như cảm biến nhiệt độ, hình ảnh từ Webcam, vận tốc của động cơ…
* Giao tiếp với các thiết bị ngoại vi thông qua nhiều chuẩn giao tiếp như: RS232, RS485, USB, PCI, TCP/IP, Enthernet.
* Mô phỏng và xử lý các tín hiệu thu nhận được để phục vụ các mục đích nghiên cứu hay mục đích của hệ thống mà người lập trình mong muốn.
* Xây dựng các giao diện người dùng một cách nhanh chóng và thẩm mỹ hơn nhiều lần so với các ngôn ngữ như VB, Matlab, Visual C…
* Cho phép kết hợp với nhiều ngôn ngữ truyền thống như C, C++…
* Cho phép thực hiện các thuật toán điều khiển như PID, Logic mờ (Fuzzy).

### 1.1.4. Các công cụ liên kết với LabVIEW

Một phần mềm lập trình phát triển cần đòi hỏi khả năng liên kết với các công cụ hỗ trợ hay thậm chí liên kết được với các phần mềm khác để có thể tối ưu một cách tốt nhất, giúp cho việc ứng dụng vào các lĩnh vực khác nhau trên thế giới, đặc biệt đó là ngành khoa học kỹ thuật. Một phần mềm ứng dụng không thể hỗ trợ được tất cả các yêu cầu của con người, vì vậy sự liện kết giữa các phần mềm là điều cần thiết. Dưới đây là một trong những công cụ hay phần mềm lập trình khác có thể liên kết được với phần mềm LabVIEW.

* Wolfram Research Mathematica.
* Microsoft Excel.
* The MathWorks MATLAB and Simulink.
* MathSoft MathCAD.
* Electronic Workbench MultiSim.
* Texas Instruments Code Composer Studio.
* Ansoft RF circuit design software.
* Microsoft Access.
* Microsoft SQL Server.
* Oracle.

### 1.1.5. Các giao thức kết nối được sử dụng trong LabVIEW

Ngoài những công cụ phần mềm liên kết với LabVIEW ra thì chúng ta cũng cần phải quan tâm đến các giao thức kết nối khác. Hiện nay, đã có rất nhiều chuẩn giao thức được xây dựng lên nhằm tối ưu khả năng chuyền tải dữ liệu một cách tốt nhất cùng với việc đáp ứng như cầu linh hoạt trong các mô hình hệ thống mà chúng ta đang xây dựng. Vì vậy, phần mềm LabVIEW đã được xây dựng các chuẩn giao thức kết nối để có thể mở rộng phạm vi ứng dụng cho các hệ thống thực tế hay những bài toán đơn giản. Dưới đây, là những giao thức kết nối được sử dụng trong LavVIEW:

* Ethernet
* CAN
* DeviceNet
* USB
* IEEE 1394
* RS-232
* GPIB
* RS-485

### 1.1.6. Các Module và các Toolkit hỗ trợ

#### 1.1.6.1. Các module chính

Để tăng cường sức mạnh và mở rộng khả năng của bộ phần mềm phát triển LabVIEW, NI cung cấp thêm các module hỗ trợ đến nhiều loại phần cứng nhúng khác nhau:

* Module thời gian thực (LabVIEW Real-Time Module).
* Module FPGA.
* Module điều khiển giám sát và ghi dữ liệu (LabVIEW Datalogging and Supervisory Control Module).
* Module biểu đồ trạng thái (LabVIEW Statechart Module).
* Module mô phỏng và thiết kế bộ điều khiển (LabVIEW Control Design and Simulation Module).
* Module phát triển thị giác (NI Visioni Development Module).
* Module cho màn hình cảm ứng và PDA (LabVIEW PDA and LabVIEW Touch Panel Module).
* LabVIEW DSP Module ( xử lý tín hiệu số ).

#### 1.1.6.2. Các Toolkit hỗ trợ người dùng

NI cũng thêm vào LabVIEW các bộ công cụ để đem lại các tiện ích khác nhau như: tạo báo cáo, phân tích nâng cao, thông tin liên lạc cơ sở dữ liệu, phân tích âm thanh và rung động. Dưới đây là một số các Toolkit hỗ trợ cho người dùng có thể sử dụng.

* Bộ công cụ kết nối cơ sở dữ liệu (LabVIEW Database Connectivity Toolkit).
* Bộ công cụ xử lý tín hiệu nâng cao (LabVIEW Advanced Signal Processing Toolkit).
* Bộ đo lường âm thanh và rung động (LabVIEW Sound and Vibration Measurement Suite).
* Bộ công cụ nhận dạng hệ thống (LabVIEW System Identification Toolkit).
* Bộ công cụ tương tác mô phỏng (LabVIEW Simulation Interface Toolkit).
* Bộ công cụ theo dõi thực thi thời gian thực (LabVIEW Real-Time Execution Trace Toolkit).
* Bộ công cụ kết nối Internet (LabVIEW Internet Toolkit).
* Bộ công cụ điều biến (LabVIEW Modulation Toolkit).
* Bộ công cụ điều khiển PID (LabVIEW PID Control Toolkit).
* Bộ công cụ thiết kế bộ lọc số (LabVIEW Digital Filter Design Toolkit).
* Bộ công cụ thu thập và xử lý hình ảnh (LabVIEW Vision Acquisition Software toolkit).

### 1.1.7. Làm việc với LabVIEW

LabVIEW được biết đến như là một ngôn ngữ lập trình với khái niệm hoàn toàn khác so với các ngôn ngữ lập trình truyền thống như ngôn ngữ C, Pascal… Bằng cách diễn đạt cú pháp thông qua các hình ảnh trực quan trong môi trường soạn thảo có sẵn hàng ngàn thư viện, hàm và cấu trúc lập trình, LabVIEW đã được gọi với tên khác là lập trình G (viết tắt của Graphical).

Những chương trình LabVIEW được gọi là những thiết bị ảo (Virtual Instruments – VIs), bởi vì hình dạng và cách hoạt động giống với những thiết bị vật lý, chẳng hạn như máy nghiệm dao động, máy hiện sóng…

Trong LabVIEW, ta xây dựng giao diện người dùng bằng cách sử dụng một bộ các công cụ và đối tượng, và cửa sổ **Front panel** được xem như là giao diện người dùng. Còn cửa sổ **Block diagram** chứa các hàm thao tác là các biểu tượng đồ họa, nơi mà dòng dữ liệu thực thi.

## 1.2. Các thành phần chính của LabVIEW

LabVIEW bao gồm các thư viện của các hàm chức năng và các công cụ phát triển được thiết kế đặc biệt dành cho thiết bị điều khiển. Các chương trình LabVIEW được gọi là những dụng cụ ảo bởi vì sự xuất hiện và hoạt động của chúng mô phỏng các dụng cụ thực tế. Các VI có cả 2 tương tác đó là: một tương tác giao diện người dùng và một mã nguồn tương đương, và truy nhập các tham số từ các VI tầng cao.

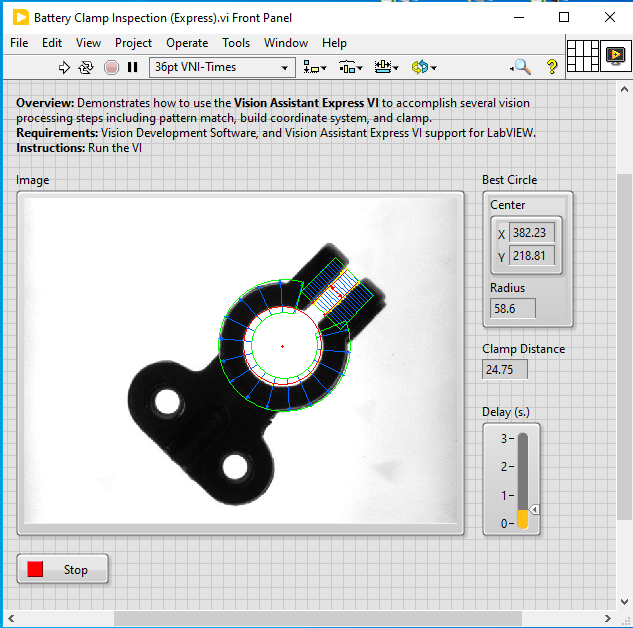
LabVIEW gồm có 3 thành phần chính đó là: bảng giao diện ( The Front Panel), sơ đồ khối (The Block Diagram) và biểu tượng & đầu nối (The Icon - Connect).

### 1.2.1. Bảng giao diện (The Front Panel)

Front Panel là giao diện mà người sử dụng hệ thống nhìn thấy. Các VI bao gồm một giao diện người dùng có tính tương tác mà được gọi là bảng giao diện, vì nó mô phỏng mặt trước của một dụng cụ vật lý. Bảng giao diện có thể bao gồm các núm, các nút đẩy, các đồ thị và các dụng cụ chỉ thị và điều khiển khác. Ta nhập vào dữ liệu sử dụng bàn phím và chuột rồi sau đó quan sát các kết quả trên màn hình máy tính.

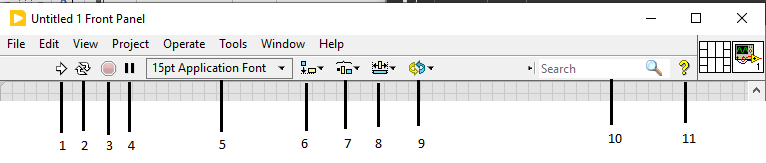
Vào **Start** >> **All Programs** >> **National Instruments LabVIEW** một cửa sổ LabVIEW xuất hiện. Ta tiếp tục chọn **Evaluate** và cửa sổ **Getting Started** sẽ xuất hiện ngay sau đó. Ta chọn **Blank VI** để hiển thị bảng giao diện hoặc ta có thể chọn **New** và sau đó hộp thoại **New** xuất hiện và trong hộp thoại đó mặc định con trở ở danh mục Blank VI. Để hiển thị bảng giao diện ta chỉ cần kích vào nút OK ở phía góc phải dưới. Cả hai cách trên đều để mở bảng giao diện mới để ta có thể xây dựng một VI mới hoàn toàn.

Ngoài ra ta có thể mở một bảng giao diện có sẵn trong LabVIEW bằng cách trong hộp thoại **New**, từ mục **Create New**, lựa chọn **VI** >> **From template** >> **Tutorial** (Getting Started) >> **Generate and Display**. Và sau đó kích nút OK để hiển thị bảng giao diện. Bảng giao diện sẽ xuất hiện như hình sau đây:



Hình 1.2: Bảng giao diện mới

Trong bảng giao diện bao gồm một thanh công cụ của các nút lệnh và các dụng cụ chỉ báo trạng thái mà ta sử dụng cho quá trình chạy và xử lý các VI. Nó cũng bao gồm những tuỳ chọn phông và các tuỳ chọn phân phối và sắp thành hàng cho việc soạn thảo các VI.



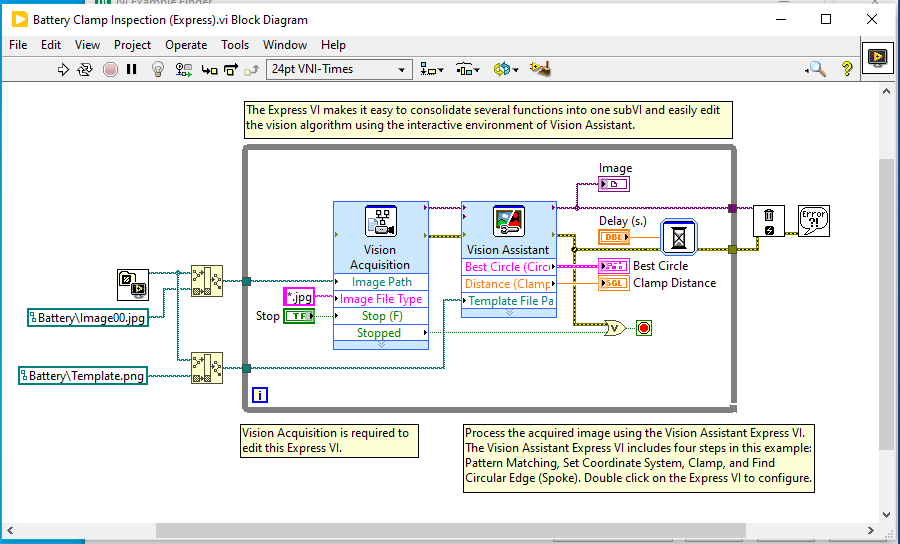
Hình 1.3: Thanh công cụ giao diện

Trong đó:

1. Nút chạy chương trình ( thanh không sáng – bị vỡ: lỗi, phải sủa lại chương trình )
2. Nút chạy nặp.
3. Nút dừng cưỡng ép chương trình.
4. Nút tạm dừng
5. Text setting (màu sắc, định dạng, kích thước - phông)
6. Gióng đều đối tượng theo hàng dọc và ngang
7. Phân bố các đối tượng.
8. Thay đổi kích thước các đối tượng
9. Lệnh bổ sung
10. Thanh tìm kiếm
11. Cửa sổ trợ giúp

### 1.2.2. Sơ đồ khối (The Block Diagram)

Sơ đồ khối chứa đựng mã nguồn đồ thị, thường biết như là mã G hoặc mã sơ đồ khối, cho đến VI chạy như thế nào. Mã sơ đồ khối sử dụng đồ thị biểu diễn các chức năng để điều khiển các đối tượng trên giao diện. Các đối tượng trên giao diện xuất hiện như biểu tượng các thiết bị trên sơ đồ khối. Kết nối điều khiển và các đầu của dụng cụ chỉ thị tới Express VIs, VIs, và các chức năng. Dữ liệu chuyển thông qua dây dẫn từ các điều khiển đến các VI và các hàm chức năng, từ các VI và các hàm chức năng đến các VI và các hàm chức năng khác, và từ các VI và các hàm chức năng đến các dụng cụ chỉ thị. Sự di chuyển của dữ liệu thông qua các nút trên sơ đồ khối xác định mệnh lệnh thực hiện của các VI và các hàm chức năng. Sự di chuyển dữ liệu này được biết như lưu đồ lập trình.



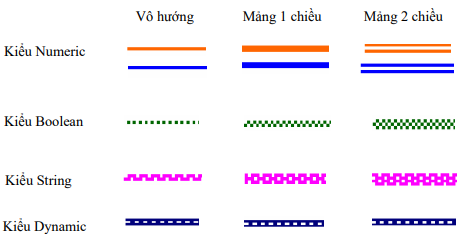
Hình 1.4: Sơ đồ khối của LabVIEW

a) Mở sơ đồ khối của một hệ thống nào đó bằng cách chọn **Window** >> **Show Block Diagram**. Hoặc cũng có thể gọi tới sơ đồ khối bằng cách trên bảng giao diện nhấn . Ví dụ Ssơ đồ khối có hình 1.4.

b) Sự phân cấp

Sức mạnh của LabVIEW định vị trong bản chất sự phân cấp của các VI. Sau khi ta tạo ra một VI, ta sử dụng nó như một VI con trong sơ đồ khối của một VI tầng cao hơn. Ta có thể có một số vô tận bản chất của các tầng trong sự phân cấp.

c) Các dạng dây nối trên sơ đồ khối



Hình 1.5: Các dạng dây nối trên sơ đồ khối

## 1.3. Những công cụ hỗ trợ lập trình labVIEW

Các công cụ lập trình trên LabVIEW bao gồm các công cụ để tạo ra các thiết bị ảo. Nó bao gồm các công cụ trong bảng giao diện (The Front Panel) và các công cụ trong sơ đồ khối (Block Diagram).

### 1.3.1 Tools Palette

LabVIEW sử dụng một bảng Tools nổi, bảng mà ta có thể sử dụng để soạn thảo và gỡ lỗi các VI. Ta sử dụng phím tới bảng thông qua các công cụ sử dụng thông thường trên bảng mẫu. Nếu ta có đóng Tools palette, chọn **View** >> **Show Tools Palette** để hiển thị bảng mẫu. Tools palette được minh hoạ như hình dưới đây:



Hình 1.6: Hình Tools Palette

 Automatic Selection Tool: công cụ lựa chọn tự động

 Operating tool: đặt những mục bảng mẫu Controls và Function trên bảng giao diện và sơ đồ khối.

 Positionting tool: những lựa chọn vị trí, thay đổi kích thước và lựa chọn các đối tượng.

 Labeling tool: soạn thảo văn bản và tạo ra các nhãn tự do.

 Wiring tool: nối dây các đối tượng với nhau trong sơ đồ khối.

 Object pop-up menu tool: mang lên trên một thực đơn pop-up cho một đối tượng.

 Scroll tool: cuộn xuyên qua cửa sổ không sử dụng thanh công cụ cuộn.

 Breakpoint tool: thiết đặt các điểm dừng trên các VI, các hàm chức năng, các vòng lặp, các chuỗi và các trường hợp.

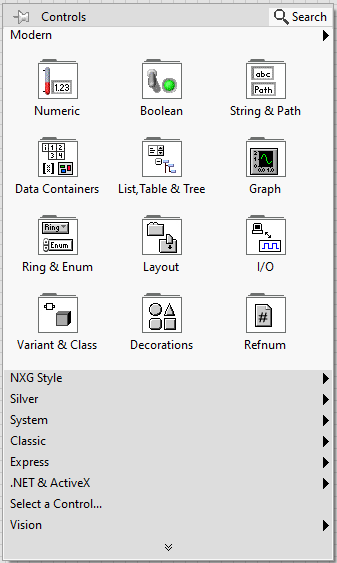
 Probe tool: tạo ra các đầu dò trên các dây.

 Color copy tool: sao chép các màu để dán tới Color tool.

 Color tool: thiết đặt các màu nền và màu nổi.

### 1.3.2. Bảng điều khiển (Controls Palette)

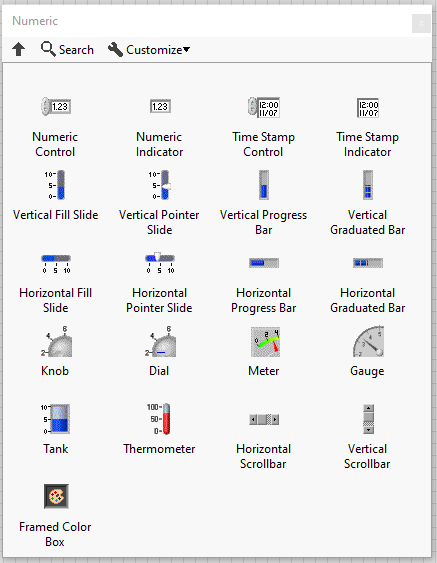
Bảng Controls bao gồm một đồ thị, bảng nổi mà tự động mở ra khi ta khởi động LabVIEW. Ta sử dụng bảng này để đặt các điều khiển và các dụng cụ chỉ thị trên bảng giao diện của một VI. Mỗi biểu tượng lớp trên chứa đựng các bảng mẫu con. Nếu bảng Controls không xuất hiện, ta có thể mở bảng bằng cách lựa chọn **View >> Show Controls Palette** từ menu của bảng giao diện. Ta cũng có thể bật lên trên một vùng mở trong bảng giao diện để truy nhập một sự sao chép tạm thời của bảng **Controls**. Sự minh hoạ sau đây hiển thị lớp đầu tiên của bảng **Controls.**



Hình 1.7: Bảng mẫu Controls

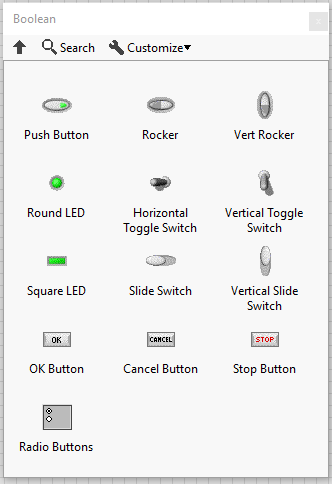
a) Numeric:

Các điều khiển và dụng cụ chỉ thị số (Numeric Controls and Indicator). Ta dùng điều khiển số để nhập các đại lượng số, trong khi đó những dụng cụ chỉ thị số thì hiển thị các đại lượng số. Hai đối tượng số được sử dụng thông dụng nhất đó là digital control - điều khiển số và digital indicator – chỉ thị số.



Hình 1.8: Bảng điều khiển và chỉ thị số

b) Boolean:



Hình 1.9: Bảng điều khiển và chỉ thị logic

Các điều khiển và dụng cụ chỉ thị kiểu logic (Boolean Controls and Indicator). Ta sử dụng điều khiển và dụng cụ chỉ thị kiểu logic cho việc nhập và hiển thị các giá trị kiểu Bool (đúng/sai- True/False). Các đối tượng đại số Bool mô phỏng các chuyển mạch - công tắc, các nút bấm, đèn LED. Các đối tượng đại số Bool được sử dụng thông dụng nhất là vertical toggle switch – công tắc đảo chiều thẳng đứng và round LED - đèn LED xung quanh.Xem ở hình 1.9.

c) Graph: Bao gồm Graph 2D, Graph 3D

Trong đó Graph 2D được chia thành 2 loại :

* Waveform graph: Dùng để biểu diễn những hàm đơn trị có dạng y = f (x), với những khoảng chia ngang nhau trên các trục.
* XY graph : dùng để biểu diễn các hàm đa trị như đường tròn hay dạng sóng thay đổi theo thời gian.

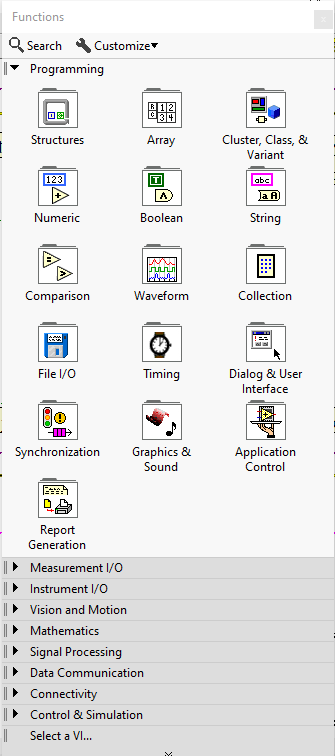


Hình 1.10: Graph

Ngoài ra, LabVIEW còn rất nhiều thư viện trong bảng mẫu Control như: System, Classic, Express, Control Design & Simulation…Trong đó có hỗ trợ rất nhiều hàm với chức năng khác nhau. Việc sử dụng các hàm trong từng thư viện hết sức linh hoạt và tùy thuộc vào mục đích sử dụng và yêu cầu từng bài toán.

### 1.3.3. Bảng các hàm chức năng (Function palette)

Bảng Function bao gồm một bảng đồ thị, bảng nổi mà tự động mở ra khi ta chuyển tới sơ đồ khối. Ta sử dụng bảng này để đặt các nút (hằng số, dụng cụ chỉ thị, các VI và …) trên sơ đồ khối một VI. Mỗi biểu tượng lớp trên chứa đựng các bảng mẫu con. Nếu bảng Function không xuất hiện rõ ràng, ta có thể chọn **View >> Show Function Palette** từ menu của sơ đồ khối để hiển thị nó. Ta cũng có thể mở ra trên một vùng mở trong sơ đồ khối để truy nhập một một sự sao chép tạm thời của bảng Functions. Lớp trên của bảng Functions được minh hoạ như hình sau đây:



Hình 1.11:Bảng Function

Việc khai thác thế mạnh của LabVIEW trên mỗi lĩnh vực phụ thuộc rất nhiều vào khả năng khai thác thư viện hàm của LabVIEW. Thư viện hàm của LabVIEW được hình tượng hoá trên bảng Funtion. Người sử dụng dễ dàng truy cập hàm cần dùng bằng cách kích chuột vào biểu tượng trên bảng.

## 1.4. Các kiểu dữ liệu

LabVIEW hỗ trợ cho tất cả các dạng dữ liệu. Các kiểu dữ liệu dạng Boolean, bytes, string, array, file, text, cluster và dạng số có thể được chuyển đổi một cách dễ dàng sang các dạng cấu trúc. Sau đây chúng ta xem xét một vài dạng dữ liệu:

### 1.4.1. Variables (biến)

Trong quá trình lập trình cần thiết phải sử dụng tới các biến. Thông qua các biến, người lập trình có thể thực hiện được việc xử lý và thay đổi dữ liệu một cách thuận lợi. Trong LabVIEW, các biến được sử dụng dưới 2 dạng là biến toàn cục (global variables) và các biến địa phương (local variables).

Global variables (biến toàn cục): Biến toàn cục được sử dụng để thực hiện công việc truyền và lưu trữ dữ liệu giữa một vài VI. Biến toàn cục được coi là một đối tượng trong LabVIEW. Khi ta tạo ra một biến toàn cục LabVIEW sẽ tạo ra một “VI toàn cục” đặc biệt. Để tạo ra các biến toàn cục, ta lựa chọn chúng trên menu **“Structs and Constants function”** sau đó đặt chúng lên Diagram. Tiếp tục cần xác định cho chúng kiểu dữ liệu thông qua các kiểu dữ liệu đã sử dụng thông qua các Controls hoặc các Indicators. Chúng ta cần chú ý là đối với mỗi biến toàn cục chúng ta cần phải tạo ra một VI với một tên riêng duy nhất. Đối với các biến toàn cục chúng ta cũng cần xác định chế độ chỉ cho phép ghi hoặc chỉ cho phép đọc. Đối với việc truy xuất vào biến toàn cục sẽ thực hiện rất nhanh chóng đối với các kiểu dữ liệu đơn giản như Numerics và Boolean. Tuy nhiên, khi ta sử dụng biến toàn cục để lưu trữ và xử lý các dữ liệu dưới dạng mảng (arrays), Clusters hay các sâu (string) lớn thì thời gian cũng như bộ nhớ cần thiết để xử lý chúng lại sẽ tương đối lớn. Vì nó đòi hỏi một vài dịch vụ quản lý bộ nhớ mỗi khi các biến đó gọi tới. Khi sử dụng các biến toàn cục cũng như các biến cục bộ thì một vấn đề có thể gặp phải là “sự tranh chấp dữ liệu”. Biến sử dụng trong Labview không hoàn toàn giống như trong các ngôn ngữ lập trình dòng lệnh. Việc “tranh chấp dữ liệu” xảy ra khi hai hoặc nhiều hơn các đoạn mã lệnh cùng thực hiện song song cùng thay đổi giá trị của một biến. Đầu ra của VI phụ thuộc vào thứ tự được thực thi của các dòng lệnh. Bởi vì nếu không có sự phụ thuộc dữ liệu giữa các biểu thức khác nhan thì sẽ không xác định được cái nào chạy trước. Nếu sử dụng các biến toàn cục trong Vis mà được thực hiện song song, thì ta có thể sử dụng một biến toàn cục thêm vào để xác định khi nào dữ liệu được phép thay đổi.

Local variable**:** cho phép người sử dụng đọc hoặc viết thông tin lên một trong những thiết bị điều khiển hoặc thiết bị hiển thị trên Front Panel. Để tạo một biến địa phương, chọn **Local Variable** từ bảng **Structs & Constant**.

Các biến địa phương bộ chỉ có tác dụng duy nhất trên các thiết bị điều khiển hoặc thiết bị hiển thị mà cùng ở trên một lược đồ. Ta không thể sử dụng biến địa phương để truy cập tới một điều khiển mà không cùng trên một lược đồ.

Ta có thể có rất nhiều các biến địa phương cho mỗi thiết bị điều khiển hoặc thiết bị hiển thị. Tuy nhiên điều đó có thể gây ra sự phức tạp, cho rằng điều khiển của ta thay đổi trạng thái một cách khó hiểu bởi vì ta ngẫu nhiên lựa chọn sai các phần tử trong một hoặc nhiều biến địa phương.

Giống như biến toàn cục, ta có thể sử dụng biến địa phương không có một “dòng dữ liệu” hợp lệ khác sử dụng.

### 1.4.2. String

Kiểu string (chuỗi) là một kiểu dữ liệu dạng chuỗi ký tự. Ta có thể sử dụng chuỗi tham gia tính toán xử lý. Để lựa chọn các ô text lưu trữ dữ liệu kiểu string ta chọn từ ô **“String & path”** từ **Control palette**.

Không chỉ vậy có thể thực hiện việc chuyển đổi từ dạng string sang các dạng khác như dạng số, dạng mảng (array). Trong LabVIEW, đôi khi người sử dụng có thể cần phải tạo ra hoặc hiển thị một số ký tự trong bảng mã ASCII mà không thể hiển thị được như: Esc, tab… Để giải quyết vấn đề đó, Labview đã cung cấp một số từ đại diện cho các từ đó, khi cần thể hiện các ký tự đó ta chỉ cần gõ vào những từ đại diện cho chúng trên Front Panel. Sau đây là một số từ thay thế (Escape Code).

### 1.4.3. Array

Trong việc lập trình đôi khi chúng ta cần xử lý một số các dãy số mà mỗi phần tử trong đó có thể được xử lý như từng thành phần riêng biệt, vì vậy mà ta cần tới mảng. Các mảng ở đây có thể là mảng một chiều (một cột hoặc một véc tơ), mảng 2 chiều, mảng 3 chiều. LabVIEW hỗ trợ cho người lập trình có thể tạo ra các mảng của mình mà trong đó chứa dữ liệu kiểu numberic, string, boolean… và rất nhiều dạng dữ liệu khác. Các mảng thường được tạo ra bởi các vòng lặp. Việc sử dụng vòng lặp là tốt cho các ứng dụng bởi vì nó xác định một vùng bộ nhớ xác định từ khi nó bắt đầu. Nếu không dùng các vòng lặp thì LabVIEW không có cách nào biết được khi nào vòng lặp lại được lặp lại, việc quản lý bộ nhớ có thể sẽ được gọi lại nhiều lần và làm chậm việc xử lý.

Người sử dụng có thể sử dụng chức năng xây dựng mảng. Nó còn cho phép ràng buộc mảng gốc vào các mảng khác.

a) Array Constant

Sử dụng hàm này sẽ tạo ra một mảng với các phần tử là các hằng số. Để định nghĩa dạng dữ liệu của mảng hằng số ta lựa chọn bất kỳ hằng số vô hướng nào từ bảng **Function Palette** và đặt nó vào bên trong mảng hằng số. Tất cả các thành phần của mảng sẽ có dạng này. Sử dụng **Operating Tool** để nhập giá trị cho từng thành phần của mảng. Ta không thể thay đổi được giá trị của mảng trong khi VI chạy.

b) Array Max & Min

Hàm này có chức năng tìm kiếm số lớn nhất và số nhỏ nhất của các số trong mảng. Hàm này thường trả về kết quả để hiển thị.

Mảng có thể là mảng n chiều của bất kỳ loại dữ liệu nào.

### 1.4.4. Các cấu trúc điều khiển luồng chương trình

Trong bất cứ ngôn ngữ lập trình nào, ta cũng hay thường gặp và làm việc với các phần tử điều khiển luồng chương trình, đó là các cấu trúc (Structures). Các cấu trúc điều khiển luồng chương trình trong một VI có 5 cấu trúc là: For Loop, While Loop, Case Structure, Sequence Structure và Fomula Node.

Các cấu trúc đó thực hiện tự động khi dữ liệu đầu vào của chúng có sẵn và thực hiện các công việc theo ý muốn cho tới khi hoàn thành thì mới cung cấp dữ liệu tới các dây nối dữ liệu đầu ra. Tuy nhiên, mỗi cấu trúc thực hiện theo các quy tắc riêng (Sub Diagram) của nó.

SubDiagram là tập hợp của các Node, Wire và Terminal bên trong đường viền của Structure. Mỗi cấu trúc For Loop và While loop có một SubDiagram.

Cấu trúc Case và Sequence có thể có nhiều SubDiagram, nhưng chỉ có một SubDiagram có thể thực hiện tại một thời điểm. Cách xây dựng các SubDiagram cũng giống như việc xây dựng các Block diagram mức đầu.

Việc truyền dữ liệu vào và ra các Structure thông qua các Terminal mà được tự động tạo ra ở nơi dây nối đi qua đường viền của cấu trúc, các Terminal này được gọi là các đường ống (Tunel).

## 1.5. SubVI và cách xây dựng SubVI

### 1.5.1. Khái niệm SubVI

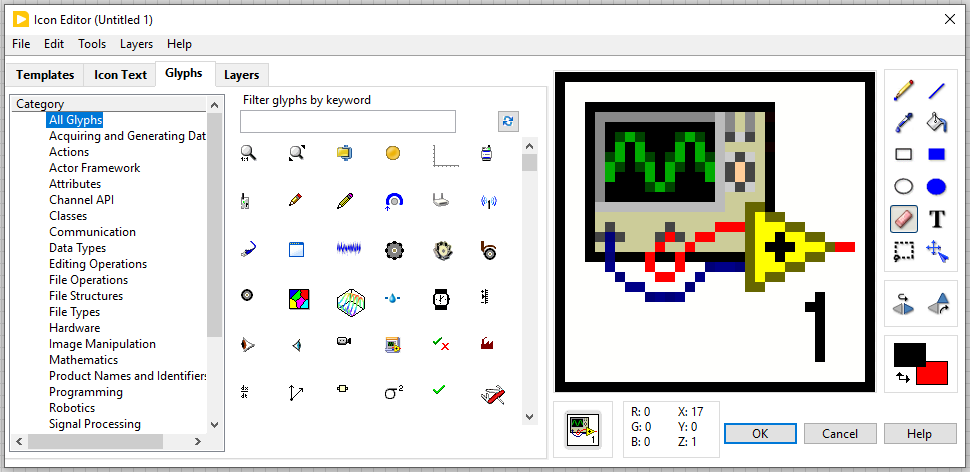
Khi thiết kế chương trình trong LabView, ta thường chú ý thiết kế các VI và xác định đầu vào và đầu ra cho chúng. Khi đó mỗi VI thực hiện một chức năng xác định. Trong việc lập trình, các VI đó có thể được sử dụng trong các Block Diagram của một VI khác ở mức độ cao hơn. Khi đó VI cao hơn đó được gọi là một SubVI. Như vậy, một SubVI ta có thể coi như một chương trình con trong các chương trình khác như C++… Trong một chương trình labVIEW, không bị hạn chế số SubVI và mỗi SubVI không bị hạn chế việc gọi đến các SubVI. Với tiện ích này, một chương trình trên Labview sẽ trở nên dễ hiểu, gọn gàng và dễ gỡ rối hơn.

### 1.5.2. Xây dựng SubVI

Có hai cách đơn giản nhất để xây dựng một SubVI: tạo một SubVI từ một VI và tạo một SubVI từ một phần của VI.

a) Tạo một SubVI từ một VI

Các SubVI được sử dụng thông qua Icon và Conector của nó. Icon của một VI là một biểu tượng đồ họa. Conetor của một VI gán các Control và Indicator cho các Terminal đầu vào và đầu ra. Muốn gọi một VI từ Block Diagram của một VI khác, thì trước tiên ta phải tạo ra Icon và Conector cho VI đó.



Hình 1.12: giao diện SubVI

Để tạo Icon, ta bấm đúp chuột trái vào biểu tượng Icon ở góc bên phía trên của Front Panel xem hình 1.13 hoặc bấm chuột phải vào biểu tượng Icon và chọn Edit Icon… Sau khi chọn Edit Icon… ta dùng các công cụ ở bên trái cửa sổ để tạo ra kiểu Icon trong vùng soạn thảo điểm lớn. Hình ảnh mặt định của Icon xuất hiện trong vùng soạn thảo. Phụ thuộc vào dạng màn hình đang sử dụng, ta có thể thiết kế kiểu Icon ở dạng đơn sắc, 16 màu hoặc 256 màu. Màu mặc định của Icon là Đen-Trắng (Black-White), nhưng nó có thể bấm vào mục chọn màu để chuyển sang 16 màu hoặc 256 màu. Ta có thể copy một Icon thành một Icon đen – trắng, và ngược lại.



Hình 1.13: Icon subVI

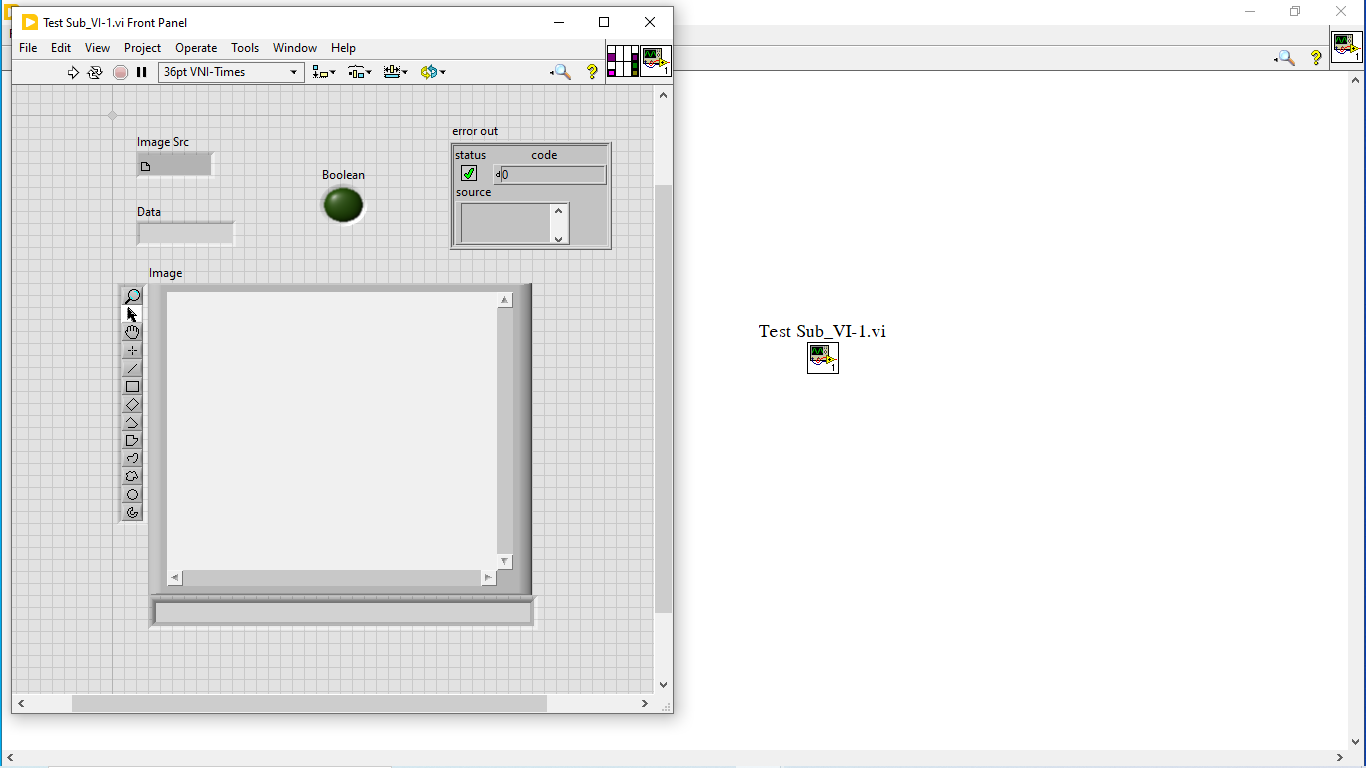
Xác định kiểu Connector Terminal: SubVI nhận dữ liệu tới và gửi dữ liệu thông qua các Terminal ở trong ô vuông Connector của nó. Ta có thể xác định các mối liên hệ bằng cách chọn số Terminal cần thiết cho VI và gán một Front Panel Control hoặc Indicator cho mỗi Terminal. Nếu Connector của VI chưa xuất hiện ở góc trái của Front Panel thì ta chọn Show Connector từ menu của Icon. Connector thay thế Icon ở góc bên phải phía trên của Front Panel. Kiểu Terminal ban đầu được chọn cho VI có số các Terminal bên phải của Connector bằng số Indicator trên Front panel và số Terminal bên trái Connector bằng số Control trên Front Panel. Nếu điều này không thể có được thì phần mềm sẽ chọn kiểu Terminal phù hợp gần nhất. Mỗi hình chữ nhật trong Connector sẽ tương ứng với một đầu vào hoặc đầu ra từ VI.

Ta có thể chọn một kiểu Terminal khác trong menu Pattern. Ta có thể thêm hoặc loại bỏ bớt Terminal bằng cách chọn Add Terminal hoặc chọn Remove Terminal. Sau khi đã chọn được Connector Terminal Pattern ta phải gán các Front Panel Control và Indicator cho các Terminal.

Bấm vào một Termianl của Connector, sau đó bấm vào một Front Panel Control hoặc Front Panel Indicator muốn gán cho Terminal đã chọn. Một khung bao xung quanh Front Panel của Control hoặc Indicator được chọn xuất hiện và nhấp nháy. Bấm vào một vùng mở của Front Panel và bấm chuột, khung nhấp nháy biến mất. Terminal và Front Panel Control hoặc Indicator đã chọn được gắn với nhau.

b) Tạo một SubVI từ một phần của VI

Để biến đổi một phần của VI thành một SubVI được gọi từ VI khác, chọn một phần trong Block Diagram của VI, sau đó chọn **Edit\Create SubVI**. Phần đó sẽ tự động thành một SubVI. Các Control và Indicator được tự động khai báo cho SubVI mới. SubVI mới tự động nối dây với các đầu dây hiện có và một Icon của SubVI sẽ thay thế phần được chọn trong Block Diagram của VI gốc. Đối với SubVI này ta có thể điều chỉnh được Icon & Connector tương tự như trên.



Hình 1.14: một chương trình subVI

# Kết luận chương 1

# Như vậy, trong chương một này, em đã đưa ra các khái quát nội dung cơ bản được tập trung tìm hiểu vào các nội dung chính sau đây :

* Các module và các toolkit hỗ trợ.
* Giới thiệu về các thành phần cơ bản chính của LabVIEW.
* Những công cụ hỗ trợ lập trình LabVIEW.
* Các kiểu dữ liệu để hỗ trợ cho việc lập trình.

# 

# 

# Chương 2: THỊ GIÁC MÁY TÍNH TRONG LABVIEW

## 2.1. Tổng quan về xử xử lý ảnh

### 2.1.1. Xử lý ảnh và quá trình phát triển

Xử lý ảnh là một ngành khoa học mới mẽ so với nhiều ngành khoa học khác nhưng tốc độ phát triển của nó rất nhanh, kích thích các trung tâm nghiên cứu, ứng dụng, đặc biệt là máy tính chuyên dụng riêng cho nó ngày càng đa dạng và mở rộng.

Con người thu nhận thông tin qua các giác quan, trong đó thị giác đóng vai trò quan trọng nhất. Những năm trở lại đây với sự phát triển của phần cứng máy tính, xử lý ảnh và đồ họa phát triển một cách mạnh mẽ và có nhiều ứng dụng trong cuộc sống. Quá trình xử lý ảnh được xem như là quá trình thao tác ảnh đầu vào nhằm cho ra kết quả mong muốn. Kết quả đầu ra của một quá trình xử lý ảnh có thể là một ảnh “tốt hơn” hay một kết luận.

Các phương pháp xử lý ảnh bắt đầu từ các ứng dụng chính như nâng cao chất lượng ảnh và phân tích ảnh. Ứng dụng đầu tiên được biết đến là nâng cao chất lượng ảnh báo được truyền qua cáp từ Luân đôn đến New York từ những năm 1920. Vấn đề nâng cao chất lượng ảnh có liên quan tới phân bố mức sáng và độ phân giải của ảnh. Việc nâng cao chất lượng ảnh được phát triển vào khoảng những năm 1955. Điều này có thể giải thích được vì sau thế chiến thứ hai, máy tính phát triển nhanh tạo điều kiện cho quá trình xử lý ảnh sô thuận lợi. Năm 1964, máy tính đã có khả năng xử lý và nâng cao chất lượng ảnh từ mặt trăng và vệ tinh Ranger 7 của Mỹ bao gồm: làm nổi đường biên, lưu ảnh.

Từ năm 1964 đến nay, các phương tiện xử lý, nâng cao chất lượng, nhận dạng ảnh phát triển không ngừng. Các phương pháp tri thức nhân tạo như mạng nơron nhân tạo, các thuật toán xử lý hiện đại và cải tiến, các công cụ nén ảnh ngày càng được áp dụng rộng rãi và thu nhiều kết quả khả quan.

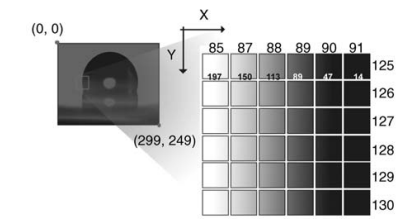
### 2.1.2. Những vấn đề cơ bản trong hệ thống xử lý ảnh.

a) Điểm ảnh (Picture Element)

Gốc của ảnh (ảnh tự nhiên) là ảnh liên tục về không gian và độ sáng. Để xử lý bằng máy tính (số), ảnh cần phải được số hoá. Số hoá ảnh là sự biến đổi gần đúng một ảnh liên tục thành một tập điểm phù hợp với ảnh thật về vị trí (không gian) và độ sáng (mức xám). Khoảng cách giữa các điểm ảnh đó được thiết lập sao cho mắt người không phân biệt được ranh giới giữa chúng. Mỗi một điểm như vậy gọi là điểm ảnh (PEL: Picture Element) hay gọi tắt là Pixel. Trong khuôn khổ ảnh hai chiều, mỗi pixel ứng với cặp tọa độ (x, y).

**Định nghĩa:** Điểm ảnh (Pixel) là một phần tử của ảnh số tại toạ độ (x, y) với độ xám hoặc màu nhất định. Kích thước và khoảng cách giữa các điểm ảnh đó được chọn thích hợp sao cho mắt người cảm nhận sự liên tục về không gian và mức xám (hoặc màu) của ảnh số gần như ảnh thật. Mỗi phần tử trong ma trận được gọi là một phần tử ảnh.

Một pixel hình ảnh thang độ xám thường chỉ có thông tin về độ sáng có thể được biểu diễn bằng các giá trị 8 bit và như vậy, hình ảnh thường được gọi là Hình ảnh 8 bit. Nếu giá trị pixel là 0, thì đó là pixel hình ảnh tối nhất (đen), trong khi giá trị 255 có nghĩa là pixel hình ảnh sáng nhất (màu trắng). Để hiểu được, Hình 2.1 cho thấy một phần được phóng đại của một hình ảnh trong đó phạm vi location của X pixel là 85–91 và phạm vi Y là 125–130 trong tổng số 300 × 250 điểm ảnh trong hình ảnh. Trong trường hợp vị trí pixel 85 dọc theo X hướng và 125 dọc theo hướng Y, giá trị pixel hình ảnh là f (85, 125) = 197, gần 255 hơn 0 và do đó được hiển thị gần với điểm cuối sáng của tỷ lệ hình ảnh (màu trắng). Mặt khác, giá trị của pixel hình ảnh trong đó X = 91 và Y = 125 là 14, gần bằng 0 và do đó tương đối tối (đen).



Hình 2.1: Ví dụ về điểm ảnh

b) Độ phân giải của ảnh.

**Định nghĩa:** Độ phân giải (Resolution) của ảnh là mật độ điểm ảnh được ấn định trên một ảnh số được hiển thị.

Theo định nghĩa, khoảng cách giữa các điểm ảnh phải được chọn sao cho mắt người vẫn thấy được sự liên tục của ảnh. Việc lựa chọn khoảng cách thích hợp tạo nên một mật độ phân bổ, đó chính là độ phân giải và được phân bố theo trục x và y trong không gian hai chiều.

Ví dụ: Độ phân giải của ảnh trên màn hình CGA (Color Graphic Adaptor) là một lưới điểm theo chiều ngang màn hình: 320 điểm chiều dọc \* 200 điểm ảnh (320\*200). Rõ ràng, cùng màn hình CGA 12” ta nhận thấy mịn hơn màn hình CGA 17” độ phân giải 320\*200. Lý do: cùng một mật độ (độ phân giải) nhưng diện tích màn hình rộng hơn thì độ mịn (liên tục của các điểm) kém hơn.

c) Mức xám của ảnh

Một điểm ảnh (pixel) có hai đặc trưng cơ bản là vị trí (x, y) của điểm ảnh và độ xám của nó. Dưới đây chúng ta xem xét một số khái niệm và thuật ngữ thường dùng trong xử lý ảnh.

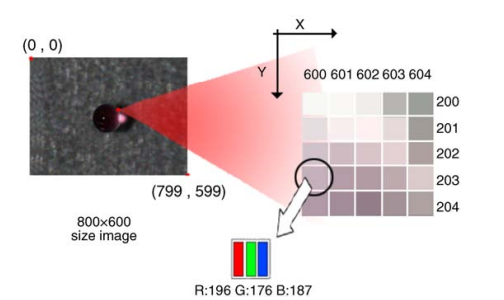
**Định nghĩa:** Mức xám của điểm ảnh là cường độ sáng của nó được gán bằng giá trị số tại điểm đó.

**Các thang giá trị mức xám thông thường:** 16, 32, 64, 128, 256 (Mức 256 là mức phổ dụng. Lý do: từ kỹ thuật máy tính dùng 1 byte (8 bit) để biểu diễn mức xám: Mức xám dùng 1 byte biểu diễn: =256 mức, tức là từ 0 đến 255).

**Ảnh đen trắng:** là ảnh có hai màu đen, trắng (không chứa màu khác) với mức xám ở các điểm ảnh có thể khác nhau.

**Ảnh nhị phân:** ảnh chỉ có 2 mức đen trắng phân biệt tức dùng 1bit mô tả 21 mức khác nhau. Nói cách khác: mỗi điểm ảnh của ảnh nhị phân chỉ có thể là 0 hoặc 1.

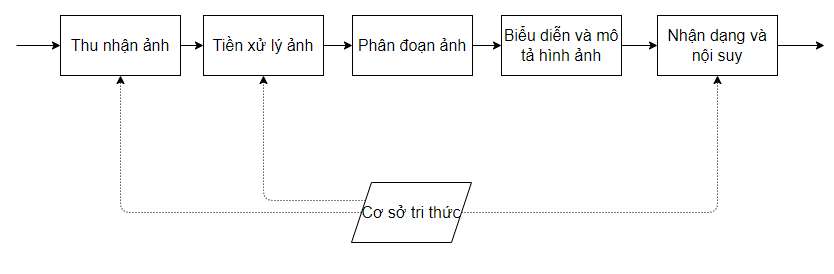
**Ảnh màu**: trong khuôn khổ lý thuyết ba màu (Red, Blue, Green) để tạo nên thế giới màu, người ta thường dùng 3 byte để mô tả mức màu, khi đó các giá trị màu: = ≈ 16,7 triệu.



Hình 2.2: Ví dụ về ảnh màu

Hình ảnh màu kỹ thuật số từ máy ảnh kỹ thuật số thường được mô tả bằng ba màu giá trị: R (đỏ), G (xanh lục) và B (xanh lam). Ba giá trị màu đại diện cho một pixel ảnh mô tả màu sắc và độ sáng của pixel. Nói cách khác, độ sáng và màu sắc của các pixel trong hình ảnh thu được từ kỷ nguyên máy ảnh màu kỹ thuật số thường được xác định bằng sự kết hợp của các giá trị R, G và B. Tất cả các màu có thể được thể hiện bằng ba màu cơ bản này. Màu kỹ thuật số hình ảnh thường được gọi là hình ảnh 24 hoặc 32 bit. Điều này là do máy tính đại diện tự nhiên của một số nguyên dưới dạng số 32 bit. *Hình 2.2*cho thấy một ví dụ về hình ảnh màu. Tổng kích thước của hình ảnh là 800 × 600. Hướng X có 800 (0–799) pixel và hướng Y có 600.

### 2.1.3. Các giai đoạn xử lý dùng trong hệ thống xử lý ảnh



Hình 2.3: Sơ đồ quá trình xử lý của hệ thống thị giác máy tính

**Phần thu nhận ảnh (Image Acquisition):** Ảnh có thể nhận qua camera màu hoặc đen trắng. Thường ảnh nhận qua camera là ảnh tương tự (loại camera ống chuẩn CCIR với tần số 1/25, mỗi ảnh 25 dòng), cũng có loại camera đã số hoá (như loại CCD – Change Coupled Device) là loại photodiot tạo cường độ sáng tại mỗi điểm ảnh.

Camera thường dùng là loại quét dòng: ảnh tạo ra có dạng hai chiều. Chất lượng một ảnh thu nhận được phụ thuộc vào thiết bị thu, vào môi trường (ánh sáng, phong cảnh**).**

**Tiền xử lý (Image Processing):** Sau bộ thu nhận, ảnh có thể nhiễu độ tương phản thấp nên cần đưa vào bộ tiền xử lý để nâng cao chất lượng. Chức năng chính của bộ tiền xử lý là lọc nhiễu, nâng độ tương phản để làm ảnh rõ hơn, nét hơn.

**Phân đoạn (Segmentation) hay phân vùng ảnh:** Phân vùng ảnh là tách một ảnh đầu vào thành các vùng thành phần để biểu diễn phân tích, nhận dạng ảnh. Ví dụ: để nhận dạng chữ (hoặc mã vạch) trên phong bì thư cho mục đích phân loại bưu phẩm, cần chia các câu, chữ về địa chỉ hoặc tên người thành các từ, các chữ, các số (hoặc các vạch) riêng biệt để nhận dạng. Đây là phần phức tạp khó khăn nhất trong xử lý ảnh và cũng dễ gây lỗi, làm mất độ chính xác của ảnh. Kết quả nhận dạng ảnh phụ thuộc rất nhiều vào công đoạn này.

**Biểu diễn ảnh (Image Representation):** Đầu ra ảnh sau phân đoạn chứa các điểm ảnh của vùng ảnh (ảnh đã phân đoạn) cộng với mã liên kết với các vùng lận cận. Việc biến đổi các số liệu này thành dạng thích hợp là cần thiết cho xử lý tiếp theo bằng máy tính. Việc chọn các tính chất để thể hiện ảnh gọi là trích chọn đặc trưng (Feature Selection) gắn với việc tách các đặc tính của ảnh dưới dạng các thông tin định lượng hoặc làm cơ sở để phân biệt lớp đối tượng này với đối tượng khác trong phạm vi ảnh nhận được. Ví dụ: trong nhận dạng ký tự trên phong bì thư, chúng ta miêu tả các đặc trưng của từng ký tự giúp phân biệt ký tự này với ký tự khác.

**Nhận dạng và nội suy ảnh (Image Recognition and Interpretation):** Nhận dạng ảnh là quá trình xác định ảnh. Quá trình này thường thu được bằng cách so sánh với mẫu chuẩn đã được học (hoặc lưu) từ trước. Nội suy là phán đoán theo ý nghĩa trên cơ sở nhận dạng. Ví dụ: một loạt chữ số và nét gạch ngang trên phong bì thư có thể được nội suy thành mã điện thoại. Có nhiều cách phân loai ảnh khác nhau về ảnh. Theo lý thuyết về nhận dạng, các mô hình toán học về ảnh được phân theo hai loại nhận dạng ảnh cơ bản:

* Nhận dạng theo tham số.
* Nhận dạng theo cấu trúc.

Một số đối tượng nhận dạng khá phổ biến hiện nay đang được áp dụng trong khoa học và công nghệ là: nhận dạng ký tự (chữ in, chữ viết tay, chữ ký điện tử), nhận dạng văn bản (Text), nhận dạng vân tay, nhận dạng mã vạch, nhận dạng mặt người…

**Cơ sở tri thức (Knowledge Base):** Như đã nói ở trên, ảnh là một đối tượng khá phức tạp về đường nét, độ sáng tối, dung lượng điểm ảnh, môi trường để thu ảnh phong phú kéo theo nhiễu. Trong nhiều khâu xử lý và phân tích ảnh ngoài việc đơn giản hóa các phương pháp toán học đảm bảo tiện lợi cho xử lý, người ta mong muốn bắt chước quy trình tiếp nhận và xử lý ảnh theo cách của con người. Trong các bước xử lý đó, nhiều khâu hiện nay đã xử lý theo các phương pháp trí tuệ con người. Vì vậy, đây các cơ sở tri thức được phát huy. Trong tài liệu, chương 6 về nhận dạng ảnh có nêu một vài ví dụ về cách sử dụng các cơ sở tri thức đó.

## 2.2. Thị giác máy tính trong công nghiệp.

### 2.2.1. Tầm quan trọng của thị giác máy tính

Từ khi được ra đời và phát triển từ những năm 50 của thế kỷ trước, máy tính điện tử liên tục được cải thiện và nâng cấp để phục vụ sự phát triển chung của nhân loại. Và chính sự phát triển của máy tính đã tạo ra những bước đại nhảy vọt trong sự tiến bộ của khoa học công nghệ toàn cầu.

Từ một chiếc máy tính nhỏ bé, người ta tạo ra vô số ứng dụng để nâng cấp và cải thiện hệ thống máy móc, dây chuyền sản xuất, quản lý chất lượng sản phẩm… Và một trong những ứng dụng đi kèm máy tính (thường là máy tính công nghiệp) làm thay đổi ngành sản xuất – phân phối và dịch vụ đó chính là ứng dụng thị giác máy tính.

**Thị giác máy tính** được ví như “*đôi mắt*” và “*bộ não*” của một máy công nghiệp. Nó không chỉ tham gia vào công tác giám sát sản phẩm trên dây chuyền mà còn tác động đến quá trình vận hành của các đơn nguyên máy trong quá trình sản xuất. Chính thị giác máy tính đã và đang góp phần giải phóng sức lao động của con người, giảm chi phí đầu vào, tăng năng suất sản phẩm công nghiệp.

Hoạt động của thị giác máy tính trong một dây chuyền sản xuất được tóm tắt như sau: các sản phẩm trên dây chuyền qua mỗi một giai đoạn sẽ được đi qua mắt thần (thường là một máy ảnh độ phân giải cao hoặc camera), thông tin số hóa từ hình ảnh của chúng sẽ được thu thập lại và gửi đến một máy tính công nghiệp để đánh giá, so sánh.

Những thông số từ hình ảnh sau khi xác định sẽ được so sánh với số liệu thành phẩm mẫu. Từ đó, các vấn đề có thể xảy ra với sản phẩm (nứt, vỡ, xước, lỗi màu…) sẽ được máy chủ phát hiện và chỉ định lệnh đã được lập trình phù hợp để xử lý sản phẩm.

Nếu các vấn đề xảy ra là một vấn đề mới chưa có trong lập trình sản xuất, máy chủ cần sự chỉ thị của con người để “*học*” những thao tác xử lý một lần duy nhất. Sau đó, khi gặp những vấn đề có tính lặp lại tương tự, hệ thống máy móc sẽ biết cách xử lý mà không cần tới sự tác động của con người.

Nhờ khả năng tự quan sát, tự giải quyết và tự “*học*” vấn đề nhờ ứng dụng thị giác máy tính, quá trình tự động hóa hoạt động đã và đang diễn ra rất mạnh mẽ và nhanh chóng.

Thị giác máy tính không chỉ tác động tích cực đến khả năng tự động hóa của các nhà máy thông minh – Smart factory – mà nó còn có những ảnh hưởng cụ thể khác đối với những lĩnh vực khác trong xã hội, cụ thể:

Hỗ trợ rất đắc lực trong việc kiểm soát độ chính xác của thành phẩm đối với lĩnh vực cơ khí chính xác hay sản xuất chi tiết điện tử. Cụ thể, thị giác máy tính hoàn toàn có thể phục dựng chính xác mô hình phôi sản phẩm với kích thước chính xác gần như tuyệt đối. Từ khối hình 3D đó, người ta sẽ xác định được những phần kích thước bị lỗi để tiến hành sửa chữa hay loại bỏ.

Thị giác máy tính còn là một trợ thủ đắc lực trong công tác nhận diện, giám sát phân loại sản phẩm – tính năng này đặc biệt quan trọng trong khối ngành thực phẩm hay dược phẩm. Chỉ cần quét qua mắt thần, các thông số về sản phẩm sẽ được xác định thông qua mã vạch trên bao bì, như vậy, việc phân loại sẽ trở nên đơn giản hơn rất nhiều.

Đối với đời sống hàng ngày, thị giác máy tính được tích hợp trong rất vật dụng có trong gia đình ta: đó là chiếc điện thoại, camera giám sát, đồ điện tử thông minh… Chúng giúp gia chủ tăng khả năng kiểm soát các thiết bị hoạt động trong nhà, đó chính là những căn nhà thông minh – smart home.

Thị giác máy tính còn được ứng dụng vào công nghệ giao thông thông minh – đó là những ứng dụng ô tô tự lái, tự biết vào làn và chấp hành đúng tín hiệu đèn giao thông. Thị giác máy tính còn tham gia vào quá trình giám sát, kiểm soát giao thông đường bộ và đường thủy bằng cách xác định và ghi nhớ các xe vi phạm thông qua ghi nhớ biển số hoặc hình ảnh của xe.

### 2.2.2. Các ứng dụng chính của thị giác máy tính trong thực tiễn

Hiện nay, thị giác máy tính là một lĩnh vực công nghệ phát triển và nó có nhiều ứng dụng rộng rãi trong đời sống thực tế, trong đó phải kể đến như:

**Phát hiện các khuyết điểm:**

Trong tất cả các ứng dụng của thị giác máy tính thì đây được xem là ứng dụng phổ biến nhất. Tính tới thời điểm hiện tại thì việc phát hiện ra những yếu tố bị lỗi sẽ được tiến hành bởi người giám sát, chỉ định và mở rộng hơn mà không thể nào kiểm soát được tất cả quy trình của hệ thống.

Nhờ có thị giác máy tính mà chúng ta có thể kiểm tra được tất cả những lỗi nhỏ nhất từ những vết nứt kim loại, tới lỗi sơn hay cả những bản in xấu có kích thước rất nhỏ, nhỏ hơn cả 0,05mm. Việc xử lý này của thị giác máy tính còn nhanh và chính xác hơn rất nhiều so với việc chúng ta sử dụng mắt thường của con người để quan sát. Thuật toán này được thiết kế với mục đích giúp cho việc ta dễ dàng phát hiện được các hình ảnh có khiếm khuyết.

**Xử lý dữ liệu:**

Các công cụ Computer Vision cùng với mô hình Deep Learning được đưa vào nghiên cứu để hỗ trợ con người thực thi nhiệm vụ nhận dạng cũng như tổ chức thông tin. Khi đó bắt buộc cần phải có một khối lượng dữ liệu lớn được dán nhãn. Các thuật toán Deep Learning phát triển và thay thế quy trình gắn thẻ thủ công thông qua phương pháp tiếp cận nghiên cứu dữ liệu đám đông. Đây là phương pháp thu thập theo thời gian thực tự động, gắn thẻ dữ liệu do các chuyên gia tạo ra để từ đó máy sẽ bắt đầu quy trình nhận dạng các đối tượng một cách dễ dàng.

**Vận hành tự động:**

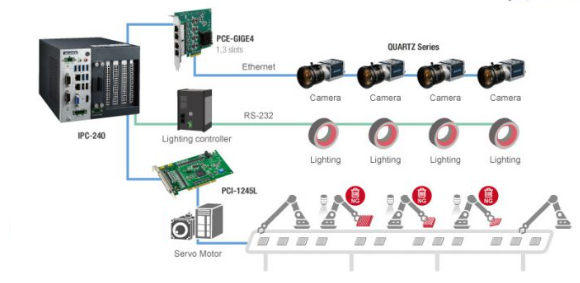
Được ứng dụng vào những chiếc xe không người lái. Đây chính là một trong những ứng dụng nổi bật của thị giác máy tính Computer vision và Deep learning. Có thể thấy những chiếc xe này chưa tới thời điểm có thể thay thế hoàn toàn người lái song công nghệ xe tự vận hành, tự lái, tự điều khiển như vậy đã có một bước tiến đáng kể hiện nay.

Bên cạnh đó với công nghệ AI thì nó đã phân tích dữ liệu được thu thập từ hàng triệu người lái xe để có thể tự động tìm làn đường và phát hiện được các mối nguy hiểm khi tham gia giao thông.

**Trình đọc tự động:**

Ta từng sử dụng hoặc nghe nói tới ứng dụng Google translate. Với ứng dụng này ta có thể phát hiện ra khả năng tìm camera từ điện thoại thông minh của ta bằng cách nhập vào văn bản với bất kỳ một loại ngôn ngữ nào và giúp dịch nó sang một ngôn ngữ khác trên màn hình một cách dễ dàng, nhanh chóng tức thì. Bằng việc sử dụng tới thuật toán nhận dạng ký tự để có thế trích xuất thông tin, điển hình là nhận dạng ký tự quang học. Đây là thuật toán cho phép dịch được một bản chính xác tiếp đó sẽ chuyển thành lớp phủ lên văn bản thực.

**Ứng dụng trong sản xuất công nghiệp:**

****

Hình 2.4: Sơ đồ mô tả hệ thống ứng dụng thị giác máy trong sản xuất



Hình 2.5: Ứng dụng thị giác máy tính trong quá trình chẩn đoán lỗi dán nhãn



Hình 2.6: Ứng dụng trong tự động hóa sản xuất ô tô công nghệ 4.0

Ngoài ra ứng dụng của Thị giác máy tính phụ thuộc rất nhiều vào ngành công nghiệp và môi trường sản xuất, một số ứng dụng điển hình cụ thể khác :

* Kiểm tra chất lượng thành phẩm sau cùng
* Kiểm tra bộ phận động cơ
* **Kiểm tra nhãn trên sản phẩm**
* Kiểm tra lỗi thiết bị y tế
* Hướng dẫn cho robot
* Xác minh mã Datamatrix
* Kiểm tra hướng của linh kiện
* Truy xuất nguồn gốc sản phẩm
* Kiểm tra bao bì
* Kiểm tra vết laser và vết cắt
* Kiểm tra lọ thuốc
* Kiểm tra gói thực phẩm
* Đọc mã vạch
* Xác minh các thành phần thiết kế

## 2.3. Các Module hỗ trợ thị giác máy tính trong LabVIEW

Để lấy hình ảnh và xử lý hình ảnh nhận trực tiếp từ Camera (như Webcam hoặc Camera công nghiệp) trong lập trình LabVIEW, chúng ta cần ít nhất 2 phần mềm ngoài phần mềm LabVIEW sau:

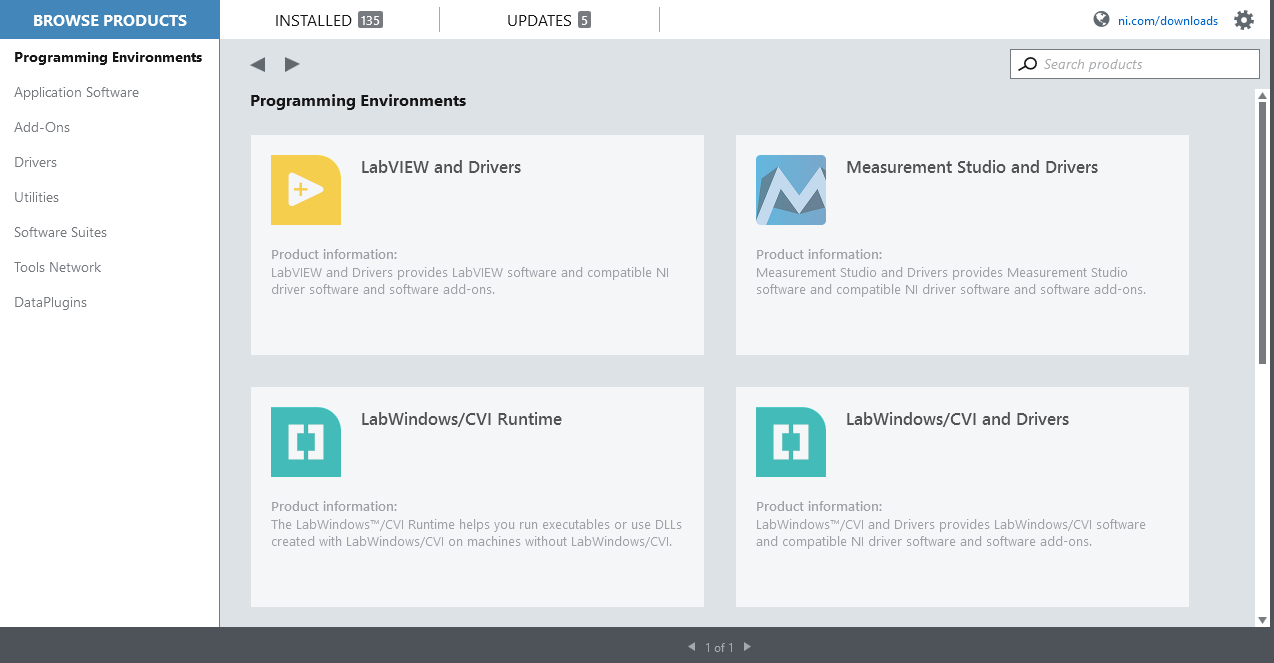
* NI Vision Development Module
* NI Vision Acquisition Software
* NI labVIEW myRIO Software Bundle
* NI-IMAQdx

### 2.3.1. NI Vision Development Module

Vision Development Module (VDM) được thiết kế để giúp ta phát triển các ứng dụng thị giác máy bằng Labview hoặc C / C ++ và triển khai các ứng dụng đó cho phần cứng Thời gian thực của Window hoặc Linux. Với thư viện chức năng toàn diện, ta có thể truy cập hàng trăm thuật toán xử lý hình ảnh và chức năng thị giác máy để nâng cao hình ảnh, kiểm tra sự hiện diện, định vị các tính năng, xác định đối tượng, đo lường các bộ phận, v.v. VDM cũng bao gồm Vision Assistant, một công cụ kỹ thuật biệt danh giúp đơn giản hóa thiết kế hệ thống thị giác bằng cách giúp ta phát triển các thuật toán để phát triển trên CPUS hoặc FPGA

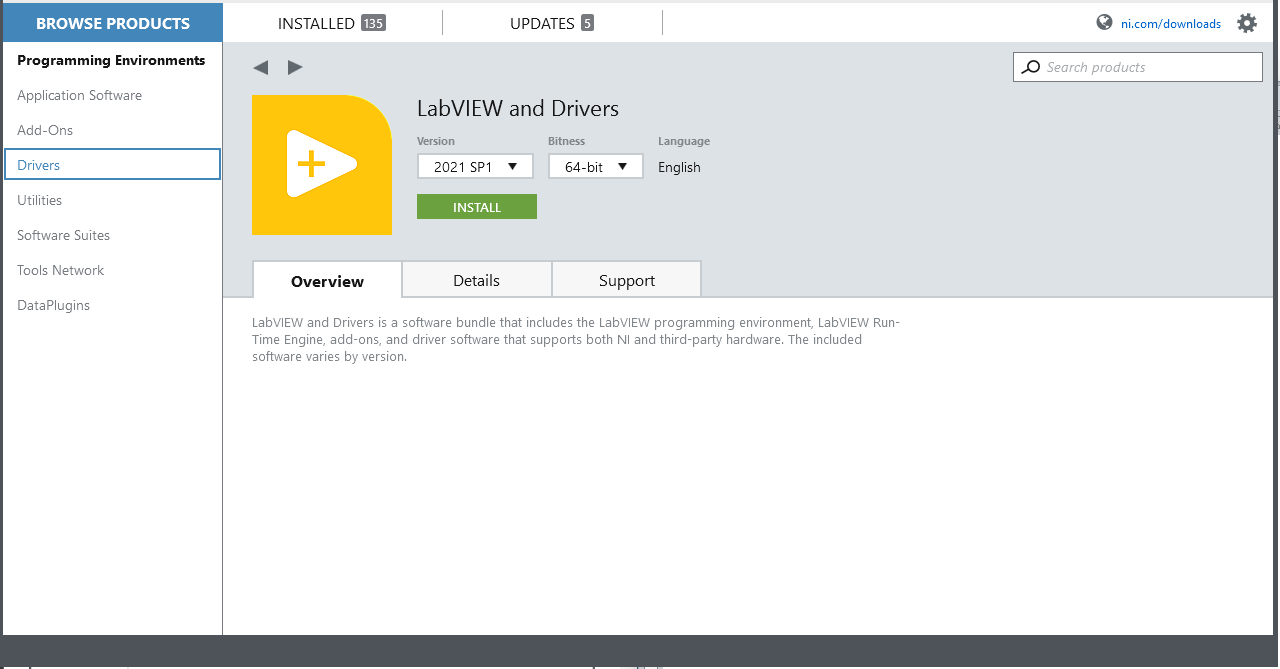
**Các bước cài đặt Module:**

Bước 1: Chạy **NI Package Mangager**, trong mục “**Programming Environment**” chọn “**LabVIEW and Drivers**”



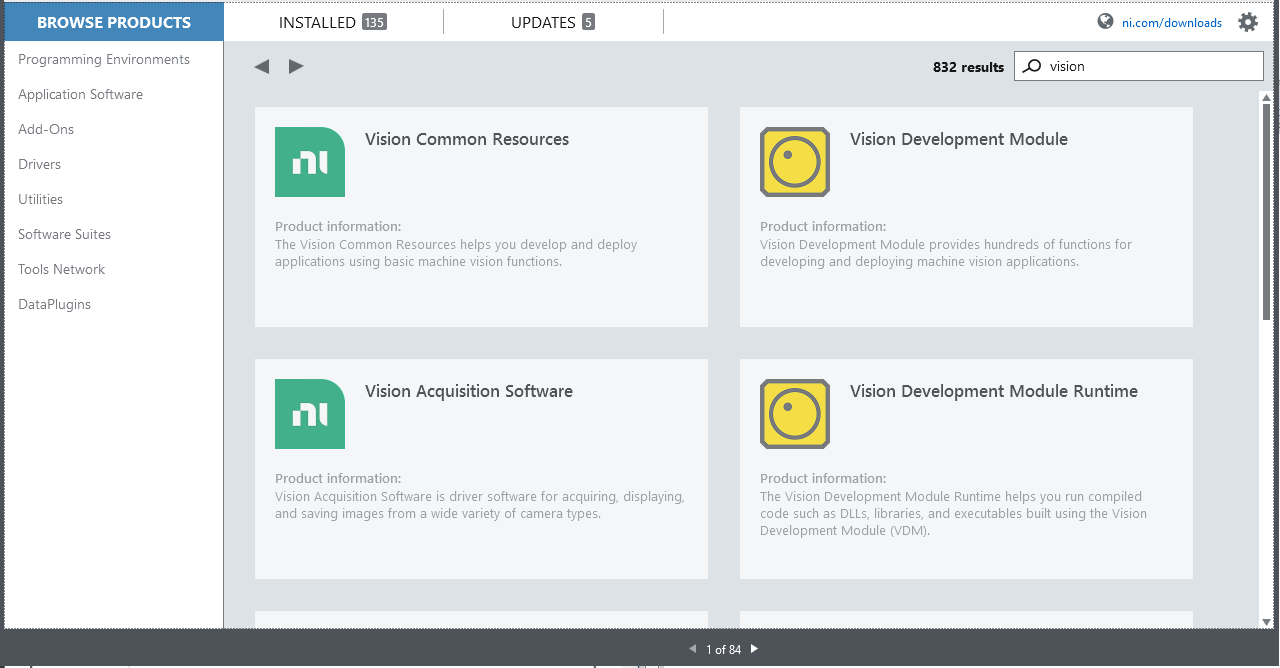
Hình 2.7: Giao diện bên trong NI Package Mangager

Bước 2: Chọn phiên bản mà mình đã cài đặt cùng với Bisness của máy tính , rồi ấn nút **Intall**.



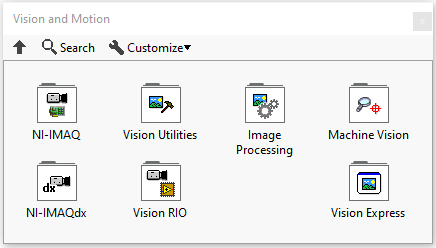
Hình 2.8: LabVIEW and Driver

Ngoài ra, ta có thể cài đặt từng phần một bằng cách tìm trên thanh tìm kiếm góc bên phải trên cùng, sau đó làm y như các bước trên đã được nêu trên.



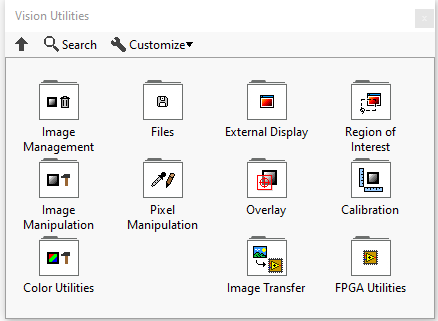
Hình 2.9: Sủ dụng thanh tìm kiếm cho Vision

Trong chương trình LabVIEW các khối chức năng thuộc Module như sau:



Hình 2.10: Vision and Motion

Vision Utilities - Các chức năng của Vision Utilities cho phép bạn thao tác và hiển thị hình ảnh trong IMAQ vision.



Hình 2.11: Các khối chức năng trong Vision Utilities

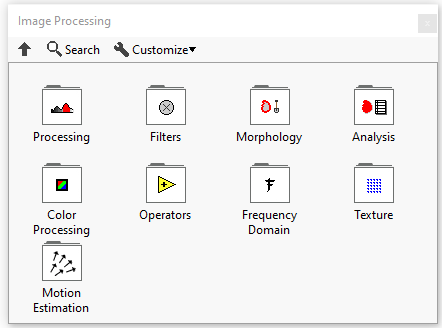
Image Management – là một nhóm VI quản lý hình ảnh. Sử dụng những VIs để tạo và xử lý hình ảnh, thiết lập và đọc các thuộc tính của hình ảnh (chẳng hạn như kích thước và độ lệch của nó), và sao chép hình ảnh này sang hình ảnh khác. Ta cũng có thể sử dụng một số VI nâng cao để xác định vùng biên giới của hình ảnh và truy cập con trỏ đến dữ liệu hình ảnh.

Files - là một nhóm VI có chức năng đọc hình ảnh từ tệp, ghi hình ảnh vào tệp ở các định dạng tệp khác nhau và nhận thông tin về hình ảnh chứa trong một tập tin.

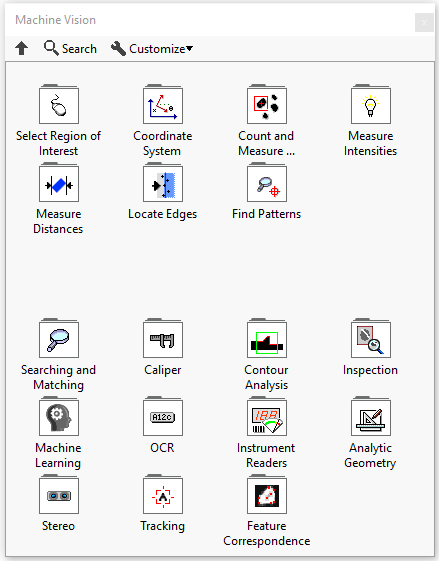
External Display – là một nhóm VI điều khiển việc hiển thị hình ảnh trong của sổ hình ảnh bên ngoài.

Region of Interest - Một nhóm VI quản lý ROI. Sử dụng các VI này để xác định ROI theo chương trình và chuyển đổi ROI sang và từ hình ảnh mặt nạ.

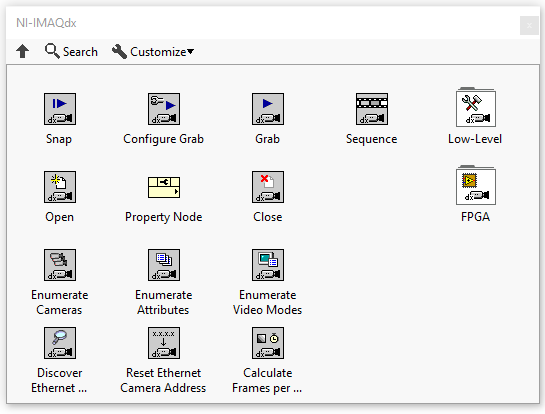
Image Manipulation - Một nhóm VI có thể sửa đổi nội dung không gian hình ảnh. Sử dụng các VI này để lấy mẫu lại hình ảnh, trích xuất các phần của một hình ảnh và xoay, dịch chuyển và mở hình ảnh. Bảng phụ này cũng chứa các VI sao chép hình ảnh vào và từ khay nhớ tạm.



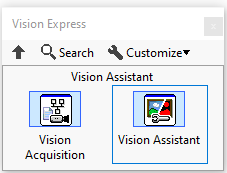
Hình 2.12: Các khối chức năng trong Image Processing



Hình 2.13: Các khối chức năng trong Machine Vision



Hình 2.14: Các khối chức năng trong NI-IMAQdx



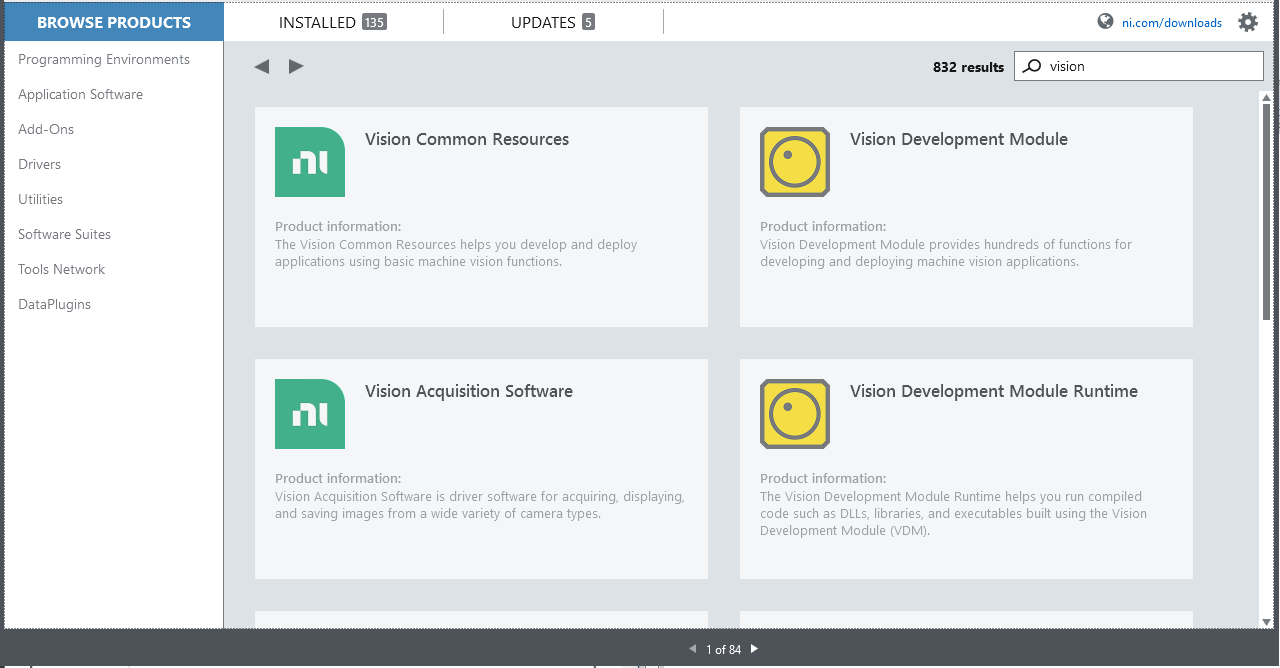
Hình 2.15: Các khối chức năng trong Vision Express

### 2.3.2. NI Vision Acquisition Software

NI Vision Acquistion Software cho phép ta thu, hiển thị và lưu hình ảnh từ một loạt các giao diện máy ảnh tiêu chuẩn công nghiệp bao gồm GigE Vision và Camera Link. Ta cũng có thể sử dụng phần mềm này để điều khiển I / O kỹ thuật số một cách thuận tiện trên phần cứng Ni vision. Với một tập hợp các chức năng và chương trình ví dụ dễ sử dụng, ta có thể nhanh chóng tạo các ứng dụng bằng LabVIEW hoặc C / C ++.

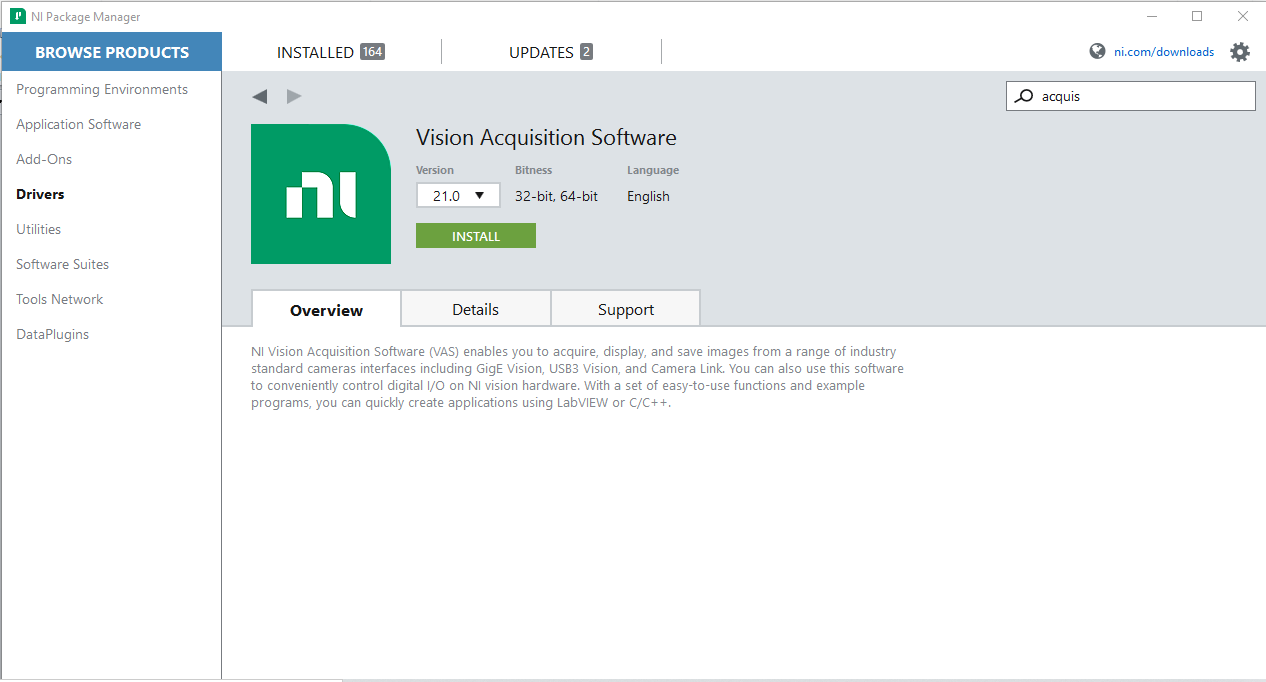
**Các bước cài đặt module:**

Bước 1: Chạy **NI Package Mangager**, trong mục “**Programming Environment**” , ấn vào thanh tìm kiếm “**NI Vision Acquisition Software**” như hình dưới đây:



Hình 2.16: Giao diện tìm kiếm NI Vision Acquisition Software

Bước 2: Chọn phiên bản mà mình đã cài đặt cùng với Bisness của máy tính , rồi ấn nút Intall.



Hình 2.17: NI Vision Acquisition Software

## 2.4. Một số kỹ thuật dùng trong thị giác máy tính cho bài toán chẩn đoán lỗi

Chúng ta đã biết đối với các lĩnh vực khác nhau đều có những giải pháp và hướng đi khác nhau. Trong lĩnh vực kỹ thuật cũng vậy, các giải pháp đều được dựa trên nghiên cứu và thực nghiệm để đưa ra những giải pháp tối ưu. Tuy nhiên, một bài toán sẽ có một hay nhiều phép tính khác nhau tùy thuộc vào các trường hợp cụ thể mà chúng ta xây dựng. Ở lĩnh vực thị giác máy tính này thì việc sử dựng nhiều thuật toán sẽ làm cho việc xử lý trở lên một cách chính xác hơn. Vì vậy, dưới đây là một số các kỹ thuật để chúng ta có thể xây dựng các giải thuật hỗ trợ cho việc xây dựng chương trình xử lý mà chúng ta mong muốn:

* Particle Analysis (Phân tích hạt)
* Edge Detection (Phát hiện cạnh)
* Pattern Matching (Khớp mẫu)
* Color Pattern Matching (Khớp mẫu màu)
* Dimension Measurement (Đo kích thước)
* Dimension Measurement Using Coordinate System
* Geometric Matching (So khớp hình học)
* Binary Shape Matching (đối sánh hinh dạng nhị phân)
* OCR (Optical Character Recognition) (Nhận dạng ký tự quang học)
* Binary Particle Classification (Phân loại hạt nhị phân)
* Contour Analysis (Phân tích đường viền)
* Image Calibration and Correction (Hiệu chỉnh và chỉnh sửa hình ảnh)
* Image subtraction (Trừ ảnh)
* Golden Template Comparison
* Image partitioning (Phân vùng ảnh)

Có rất nhiều phương pháp trong việc xử lý hình ảnh như đã được nêu ở trên. Trong đó có 3 phương pháp cơ bản nhất:

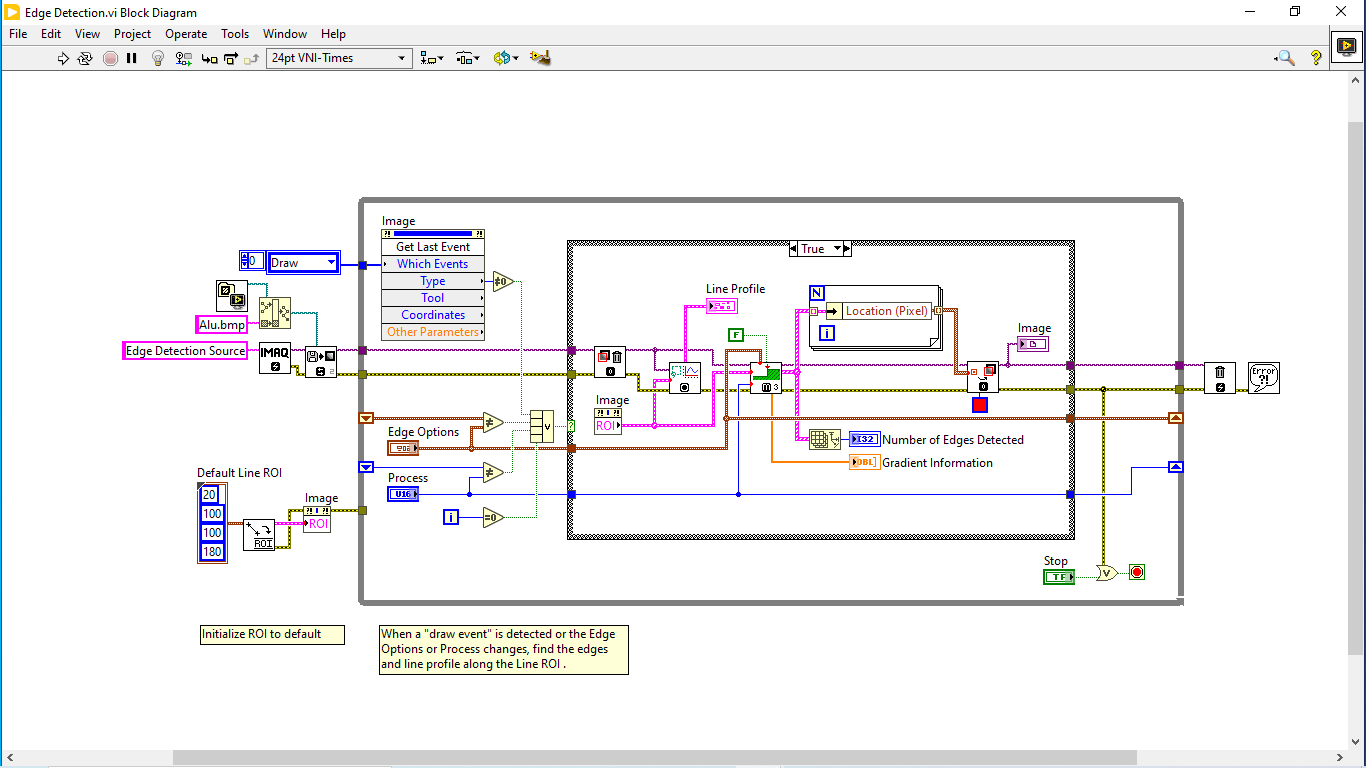
* Edge Detection
* Pattern Matching
* Image Subtraction

Được sử dụng để xử lý một số các bài toán đơn giản như đo góc lệch, độ dài, sắc định màu, tìm các khuyết điểm,đếm số lượng các vật có trong hình ảnh,…

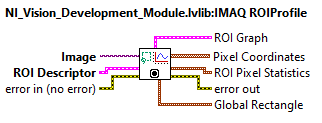
### 2.4.1. Phương pháp Edge Detection

Edge dectection (Phát hiện cạnh) được sử dụng để tìm các vị trí trong hình ảnh kỹ thuật số nơi hình ảnh độ sáng thay đổi đột ngột dọc theo một đường pixel. Những thay đổi đột ngột này thường xác định cạnh của một đối tượng trong một hình ảnh. Bằng cách sử dụng tính năng phát hiện cạnh, ranh giới của một đối tượng có thể được xác định. Khi ranh giới của một đối tượng được định vị, kích thước cũng như các đặc điểm khác của đối tượng có thể được xác định.

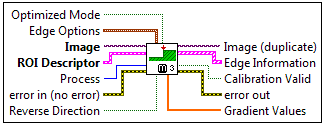
Để sử dụng thuật toán phát hiện cạnh thì hình ảnh màu thu được từ Camera (webcam) phải được chuyển đổi trước thành thang đo độ xám hoặc hình ảnh nhị phân. Ví dụ một chương trình cụ thể:



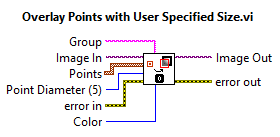
Hình 2.18: Chương trình bên Block Diagram



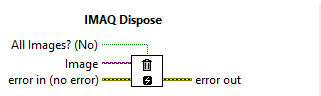
Hình 2.19: Khối IMAQ ROIProfile



Hình 2.20: Khối IMAQ Edge Tool 3



Hình 2.21: Khối Overlay Points with User Specified Size

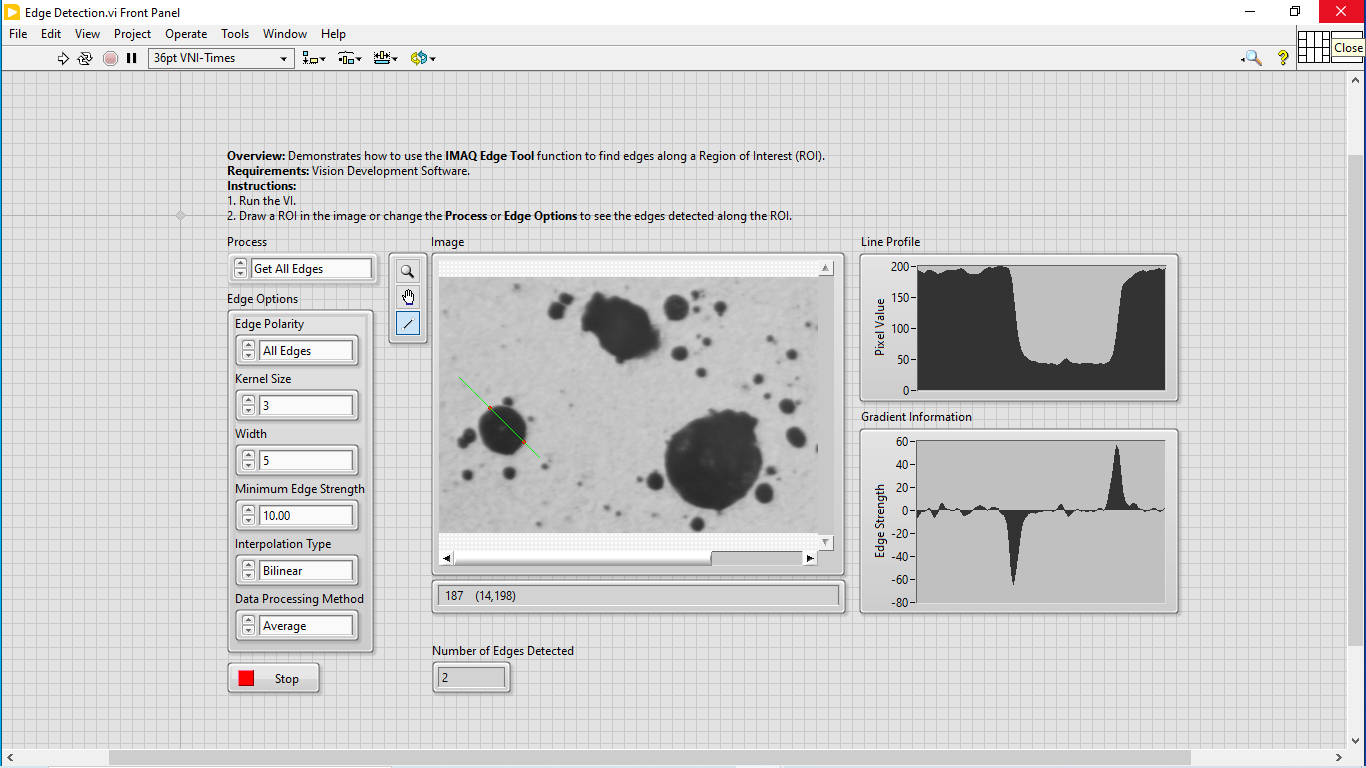


Hình 2.22: IMAQ Dispose

Nút thuộc tính ROI  của màn hình hình ảnh cung cấp thông tin về ROI. Nó chứa thông tin bao gồm vị trí và kích thước của ROI như được vẽ trên màn hình Image. Các giá trị hình ảnh dọc theo đường ROI thu được bằng cách sử dụng tệp **ROIProfile.**

Chức năng **Edge Tool** được sử dụng để tìm cạnh dọc theo đường ROI. Trả về vị trí và kết quả thống kê trong thông tin cạnh.

Overlay VI sử dụng tọa độ cạnh xử lý hình ảnh kết quả và hiển thị các vị trí này dưới dạng hình bầu dục nhỏ phủ trên hình ảnh. Overlay Points with User Specified Size VI yêu cầu các vị trí cạnh (Point) thu được bằng thuật toán phát hiện cạnh làm đầu vào. Giá trị Location (Pixel) trong cụm Edge information cung cấp những thông tin cần thiết.

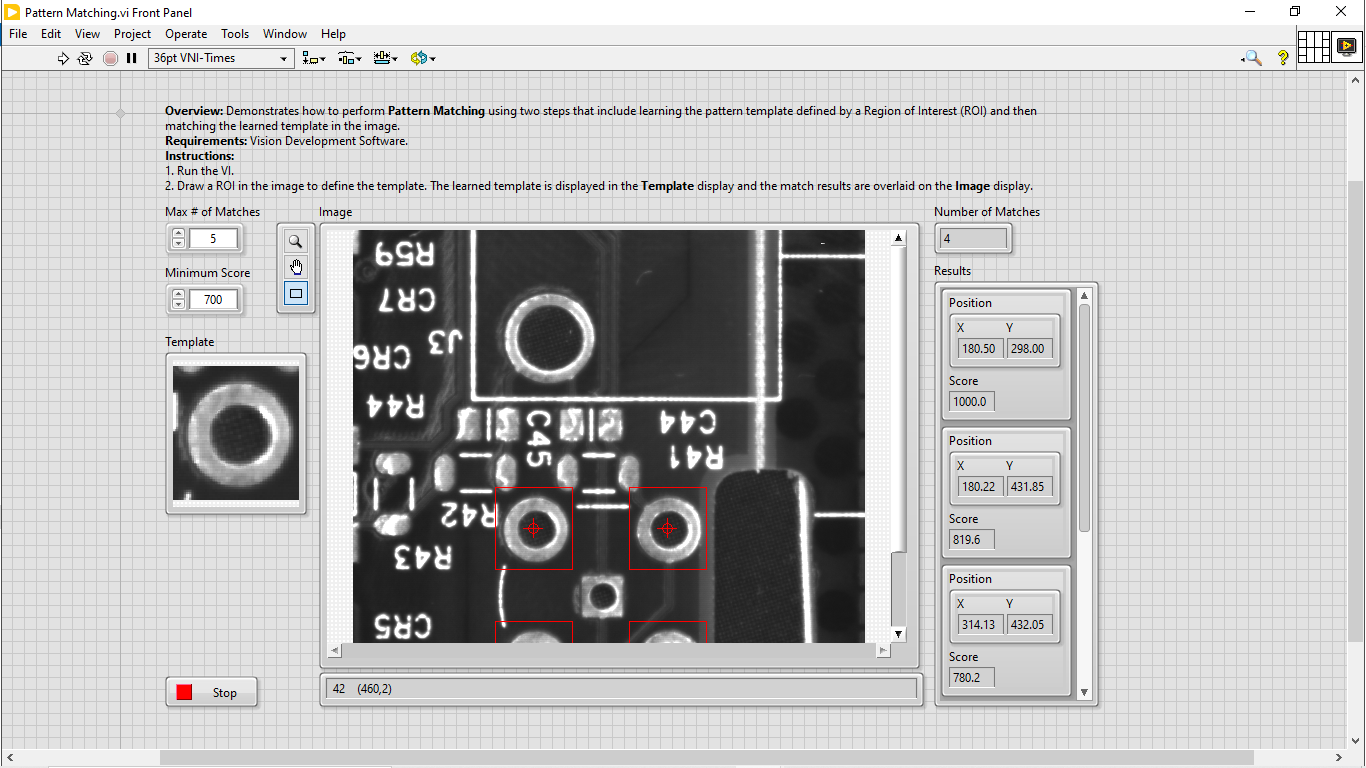


Hình 2.23: Giao diện chương trình bên Front Panel

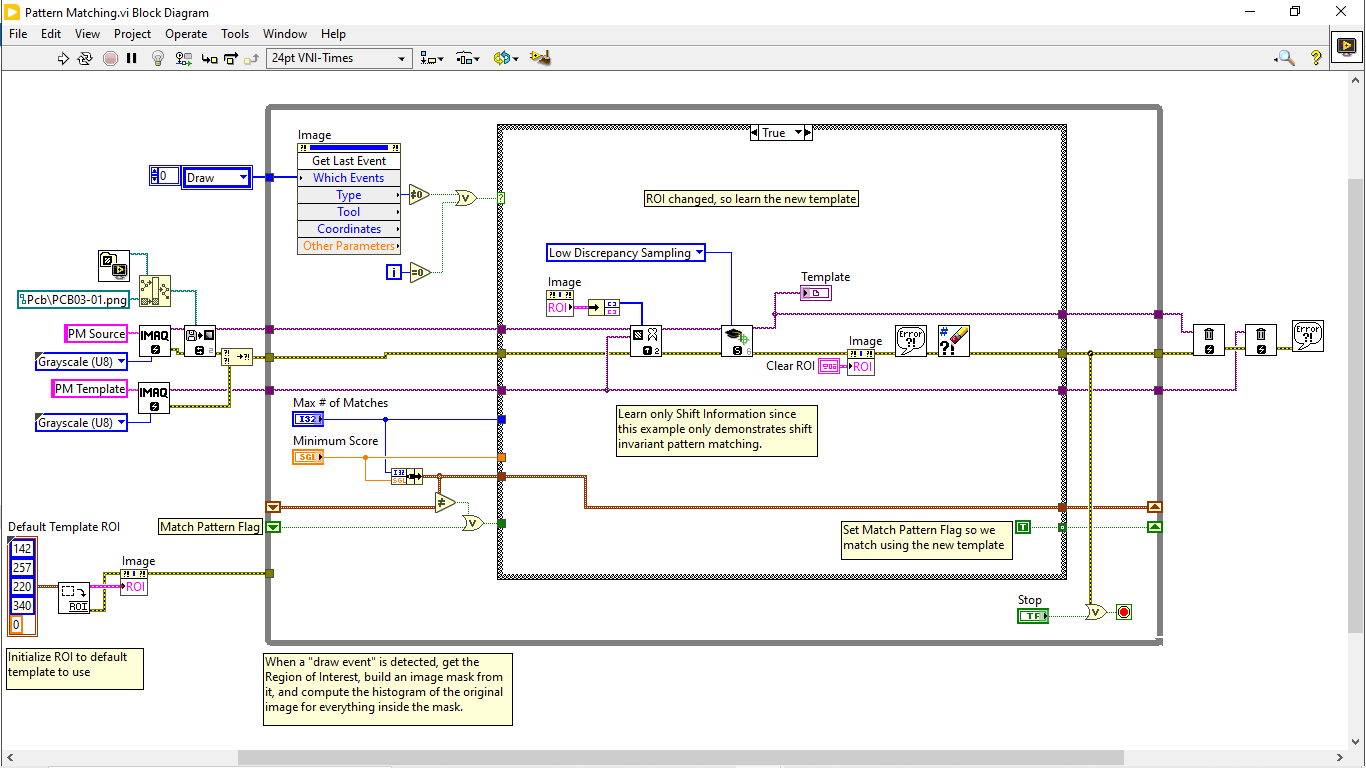
Bằng cách xác định đường ROI trên đối tượng (là đường màu xanh lá cây), các giá trị Pixel của đối tượng dọc theo đường ROI được gửi lại trong Biểu đồ cấu hình đường ở trên cùng bên phải***.*** Các chấm đỏ nhỏ ở các cạnh ROI với đối tượng cho biết rằng chương trình đã tìm thấy các vị trí cạnh dọc theo đượng ROI. Các dấu chấm xuất hiện trên hình ảnh nhờ một hàm Overlay.

### 2.4.2. Phương pháp Pattern Matching

Đối sánh mẫu là một phương pháp để tìm các vùng trong hình ảnh thang độ xám phù hợp một mẫu hình ảnh tham chiếu. Nếu nguồn ảnh ban đầu là ảnh màu thì ảnh trước tiên cần được chuyển đổi sang thang độ xám để sử dụng đối sánh mẫu. Các đối sánh mẫu VI sử dụng hình ảnh tham chiếu hoặc mẫu để tìm các hình ảnh giống như trong một hình ảnh mới bất kể vị trí, xoay hoặc tỷ lệ của mẫu. Đối sánh mẫu thường được sử dụng để xác định vị trí của một dấu tròn, hoặc các tính năng đặc trưng độc đáo của một đối tượng trong ảnh. Ta có thể sử dụng các vị trí để tính chiều dài, góc và các phép đo khác. Kết quả là, mẫu khớp đã được sử dụng rộng rãi trong các ứng dụng khác nhau như căn chỉnh, đo lường và kiểm tra. So khớp mẫu có lợi thế hơn so với phân tích hạt hoặc phát hiện cạnh bởi vì tìm kiếm mẫu không dựa vào độ sáng riêng biệt của đối tượng được chụp ảnh so với nền ảnh.

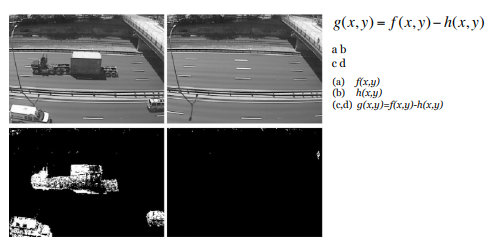


Hình 2.24: Giao diện chương trình Pattern Matching bên Front Panel



Hình2.25: Giao diện chương trình Pattern Matching bên Block Diagram

### 2.4.3. Phương pháp Image subtraction



Hình 2.26: Ảnh khi được Subtract

Ta sẽ tính hiệu giá trị Pixel của khung hình hiện tại và khung hình trước đó nếu hiệu này lớn hơn ngưỡng thì Pixel tại vị trí đó được hiện lên. Ví dụ đối với một hình ảnh cụ thể sau:



Hình 2.27: Hình ảnh mẫu



Hình 2.28: Hình ảnh có vết bẩn và hình đã được qua xử lý ảnh bằng phương pháp Suctract

# Kết luận chương 2

# Như vậy, em đã đưa ra những nội dung trọng tâm sau đây:

* Tầm quan trọng của thị giác máy tính trong công nghiệp
* Các Module và cách cài đặt các Module trong LabVIEW để hỗ trợ trong công việc xử lý ảnh để chẩn đoán lỗi nhãn dán sản phẩm
* Đưa ra các giải thuật xử lý ảnh được sử dụng rộng rãi trong thị giác máy tính

Từ những cơ sở lý thuyết trên và các công cụ đã tìm hiểu ở chương 1 và chương 2 này, em sẽ sử dụng những nội dung đó để thực hiện bài toán chẩn đoán lỗi dán nhãn sản phẩm mà sẽ được trình bày ở chương 3.

# 

# Chương 3: ỨNG DỤNG THỊ GIÁC MÁY TÍNH CHO BÀI TOÁN CHẨN ĐOÁN LỖI DÁN NHÃN SẢN PHẨM

## 3.1. Tổng quan về hệ thống xây dựng

# Trước khi vào chương trình cụ thể em xin giới thiệu hệ thống mà em dựa trên đó để xây dựng hệ thống mô hình thực nghiệm.

# 

Hình 3.1: Tổng quan về hệ thống xây dựng

### 3.1.1. Máy tính (computer)

# Máy tính hay còn gọi là máy vi tính, là hệ thống dùng để tính toán, kiểm soát các hoạt động có thể biểu diễn dưới dạng số hay quy luật logic. Máy tính được nắp ghép bởi các thành phần có thể thực hiện các chức năng đơn giản đã được định nghĩa trước. Quá trình tác động của các thành phần này đã tạo cho máy tính khả năng xử lý thông tin. Hiện nay có hai loại máy được sử dụng rộng rãi đó là máy để bàn, máy sách tay.

# 

Hình 3.2: Camputer

# 

Hình 3.3: Laptop

# Nhiệm vụ của máy tính là chứa phần mềm lập trình, hiển thị giao diện cho người dùng dễ dàng quan sát kết quả cũng như thiết lập cài đặt trực tiếp và thực hiện kết nối với thiết bị ngoại vi. Ở trong hệ thống mà em xây dựng thì máy tính đóng vai trò là thiết bị lập trình và chạy giải thuật xử lý cùng với việc hiển thị hình ảnh thu được từ Webcam.

### 3.1.2. Camera (webcam)

Webcam là viết tắt của Website Camera, đây là loại thiết bị ghi hình kỹ thuật nối với máy tính để truyền trực tiếp hình ảnh mà nó ghi được đế một máy tính khác truyền lên một website nào đó thông qua mạng Internet.

Về cơ bản, webcam gần giống với máy ảnh kỹ thuật số nhưng khác ở chỗ các chức năng chính của nó sẽ do phần mềm cài đặt trên máy tính điều khiển và xử lý.

Các dữ liệu mà webcam thu được sẽ được lưu vào máy tính, dùng như thiết bị liên kết hình ảnh giữa con người với nhau. Ngoài ra nó còn sủ dụng cho các công ty an ninh, ví dụ như truyền hình ảnh nó thu được đến trung tâm kiểm soát từ xa.

Webcam có rất nhiều công dụng trong mọi lĩnh mực đời sống như một vài lĩnh vực dưới đây:

* Phục vụ cho nhu cầu gọi điện trực tiếp: gọi điện Video trực tuyến, điều này giúp cho chúng ta có thể nhìn thấy nhau mang lại cảm giác giống như bên ngoài.
* Phục vụ cho nhu cầu chơi Game.
* Đăng nhập vào máy tính thông qua nhận diện khuân mặt.
* Phục vụ quay phim và ghi âm.
* Phục vụ nhu cầu giám sát về mặt an ninh trong các khu chung cư hay nhà riêng.
* Phục vụ trong nhu cầu sản xuất của các xí nghiệp sản xuất hiện nay bao gồm các khâu như là: Đọc mã vạch để thanh toán; nhận diện ký tự để quản lý trong các khu để xe; giám sát quá trình sản xuất, các thiết bị trong xưởng; trong các khâu chẩn đoán lỗi sản phẩm. Thậm chí nó còn được sử dụng để chế tạo các loại Robot thông minh dùng webcam làm “Mắt”.

Trong đề tài nghiên cứu này, em sử dụng loại Webcam Rapoo C270L của hãng Rapoo sản xuất, có nhiệm vụ đưa hình ảnh mà nó nhìn thấy vào máy tính để chương trình được xây dựng sẵn trên phần mềm LabVIEW xử lý có hình dưới đây:



Hình 3.4: Webcam Rapoo C270L

Thông số kỹ thuật:

* Độ phân giải : FHD 1080P/ HD 720P
* Kết nốt: USB2.0
* Hệ điều hành tương thích: Windows7/8/10 or above, Mac OS X 10.6 or above Chrome
* Kích thước sản phẩm: 50\*60\*50 mm
* Độ phân giải tối đa: 1080p/30fps (lên tới 1920 x 1080) & HD 720P (lên tới 1280 x 720)
* Loại tiêu cự: lấy nét cố định
* Phạm vi quan sát: 105 độ

### 3.1.3. Thiết bị chiếu sáng

# Thiết bị chiếu sáng là một thiết bị cực kỳ quan trọng trong lĩnh vực xử lý ảnh. Khi chúng ta không cấp đủ ánh sáng hay là cấp ánh sáng quá cao làm cho việc nhận hình ảnh của Webcam trở lên khó khăn. Vì có thể làm cho Webcam bị chói hay môi trường xung quang trở lên tối om không nhìn rõ vật.

# Vì vậy, việc cấp ánh sáng rất quan trọng, nó ảnh hướng rất lớn đến độ chính xác của hệ thống, cho lên ta cần xây dựng một hệ thống ánh sáng ổn định và ánh sáng tốt nhất là ánh sáng trắng để giúp cho việc thu nhận hình ảnh rõ nét hơn. Tuy nhiên, cùng với việc cấp ánh sáng ổn định thì điều không thể thiếu đó là tạo một vùng kín tức là không có ánh sáng nào khác ngoài ánh sáng cấp cho hệ thống, điều này tạo lên độ chính xác trong khâu cấp ánh sáng và nó cũng tăng tỉ lệ chính xác cho việc xử lý ảnh.

# Trong hệ thống mà em xây dựng thì em sử dụng 2 bóng đèn led ánh sáng trắng và kết hợp với triết áp để tùy chỉnh cường độ ánh sáng mong muốn.

## 3.2. Giải pháp chẩn đoán lỗi dán nhãn sản phẩm

### 3.2.1. Đối tượng thực nghiệm

Nhãn dán là một loại nhãn mác thường gặp và được nhiều đơn vị sản xuất hàng hóa chọn lụa. Nhãn dán là một thuật ngữ được sử dụng khá phổ thông hiện nay. Ta có thể hiểu một cách đơn giản nhãn dán chính là bản viết tay, in hoặc bản vẽ, bản chụp của chữ,… Thậm chí là hình ảnh, hình vẽ được dán hoặc in trực tiếp lên bao bì của sản phẩm.

Đối với một nhà phân phối, in nhãn dán là một trong những công đoạn quan trọng, đòi phải thực hiện một cách nghiêm túc và chăm chút nhất.

Hiện nay, việc in nhãn dán lên sản phẩm chính là cách để các nhà sản xuất phân biệt được sản phẩm của mình với những sản phẩm khác có cùng loại, cùng mẫu mã.



Hình 3.5: Một số nhãn tiêu biểu



Hình 3. 6: Hình ảnh sản phẩm lọ tương ớt Chin-Su

Trong đề tài nghiên cứu này, nhãn dán được dùng để làm sản phẩm chạy thực nghiệm là nhãn dán của lọ tương ớt Chin-Su. Trên thế giới, sản phẩm tương ớt Chin- Su đã được sản xuất qua rất nhiều mẫu mã khác nhau đi kèm với đó là hình dáng, chất lượng sản phẩm, thành phần bên trong, dẫn đền giá cả cũng khác biệt rõ ràng chúng ta có thể thấy một số hình ảnh lọ tương ớt trong Hình 3.6. Với đề tài nghiên cứu này, em sẽ sử dụng lọ tương ớt Chin-su của nhà sản xuất Masan Consumer của công ty TNHH Một thành viên Công nghệ Masan, có hình cụ thể dưới đây:

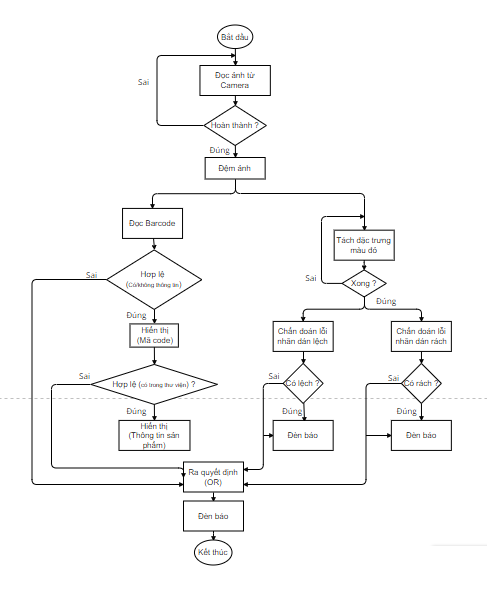




Hình 3.7: Một số hình ảnh về lọ tương ớt

### 3.2.2. Giải pháp chẩn đoán lỗi dán nhãn sản phẩm

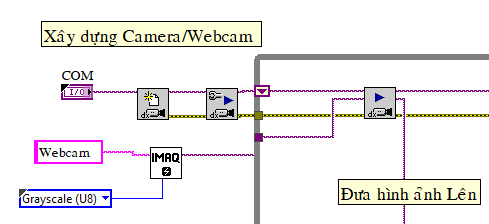
Khi xử lý một bài toán nào đó, chúng ta cần phải đưa ra hướng giải quyết một cách rõ ràng. Vì vậy, trước khi xây dựng chương trình chẩn đoán lỗi dán nhãn sản phẩm thì ta có lưu đồ thuật toán sau đây:



Hình 3.8: Lưu đồ thuật toán

#### 3.2.2.1. Khâu xây dựng kết nối với cổng Webcam

Đây là bước đầu tiên để xây dựng một chương trình xử lý ảnh trên phần mềm LabVIEW. Nó là bước quan trọng nhất không thể thiếu vì nếu thiếu nó sẽ không thể thu nhận hình ảnh thực tế để thực hiện các bài toán xử lý ảnh nâng cao được.

****

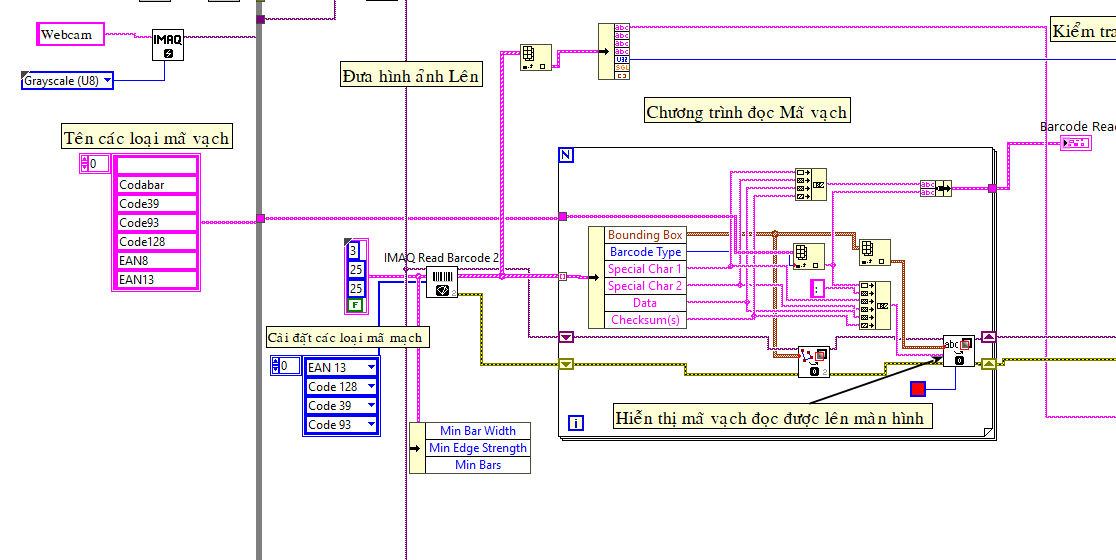
Hình 3.9: Thiết lập kết nối với Webcam

Trong việc xây dựng kết nối với Webcam thì ta sử dụng 5 khối cơ bản không thể thiếu sau đây:

* IMAQdx Open Camera VI: Chức năng là mở máy ảnh, truy vấn máy ảnh và tải tệp cấu hình máy ảnh để tạo tham chiếu duy nhất cho máy ảnh.
* IMAQdx Configure Grab VI:Chức năng là thực hiện việc cấu hình và chuyển đổi lặp đi lặp lại liên tục một vòng bộ đệm.
* IMAQ dx Grab2 VI: Chức năng là lấy khung hình mới nhất khi mà đã được cấu hĩnhong để đưa vào Image Out. Hình ảnh được lấy từ khối IMAQdx Configure Grab VI. Nếu loại hình ảnh không phù hợp với định dạng video của máy ảnh, VI này sẽ thay đổi hình ảnh thành một định dạng phù hợp.
* IMAQ Create VI: Tạo vị trí bộ nhớ tạm thời cho hình ảnh.
* IMAQdx Close Camera VI:Dừng quá trình chuyển đổi đang diễn ra, giải phóng tài nguyên liên quan đền chuyển đổi và đóng Phiên máy ảnh được chỉ định.

#### 3.2.2.2. Khâu đọc Barcode và truy xuất thư viện Barcode

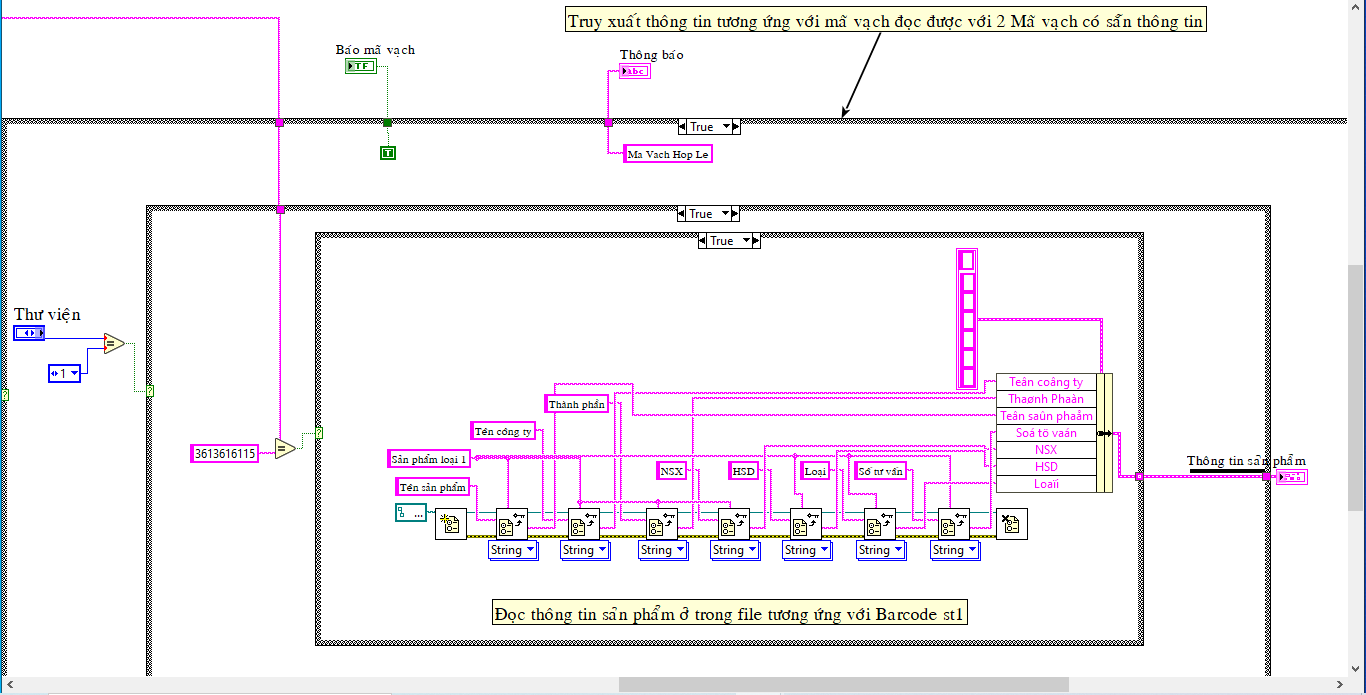
Khi Webcam được bật trình thu nhận hình ảnh và tiền xử lý đã được xây dựng ở khâu xây dựng kết nối Camera ở mục 3.2.2.1. Vì vậy, sau khi hình ảnh đã tiền xử lý qua sẽ được đưa vào khâu đọc Mã vạch ở hình 3.10 dưới đây. Hình ảnh sẽ được đưa vào **IMAQ Read Barcode 2** để khối này nhận diện mã vạch trong hình ảnh được đưa vào. Sau đó, sẽ đẩy thông tin Data của mã vạch bao gồm: số vạch, loại mã vạch, số bắt đầu và kết thúc. Các thông tin này sẽ được hiển thị trên màn hình giám sát thông qua 2 khối chức năng là **IMAQ Overlay Multiple Lines 2** và **IMAQ Overlay Text**, hai khối này có chức năng tìm 2 giá trị đầu và cuối của mã vạch và đọc ký tự trên mã vạch rồi đưa ra màn hình. Có thể xem ở hình 3.10 dưới đây.

****

Hình 3.10: Đọc Barcode

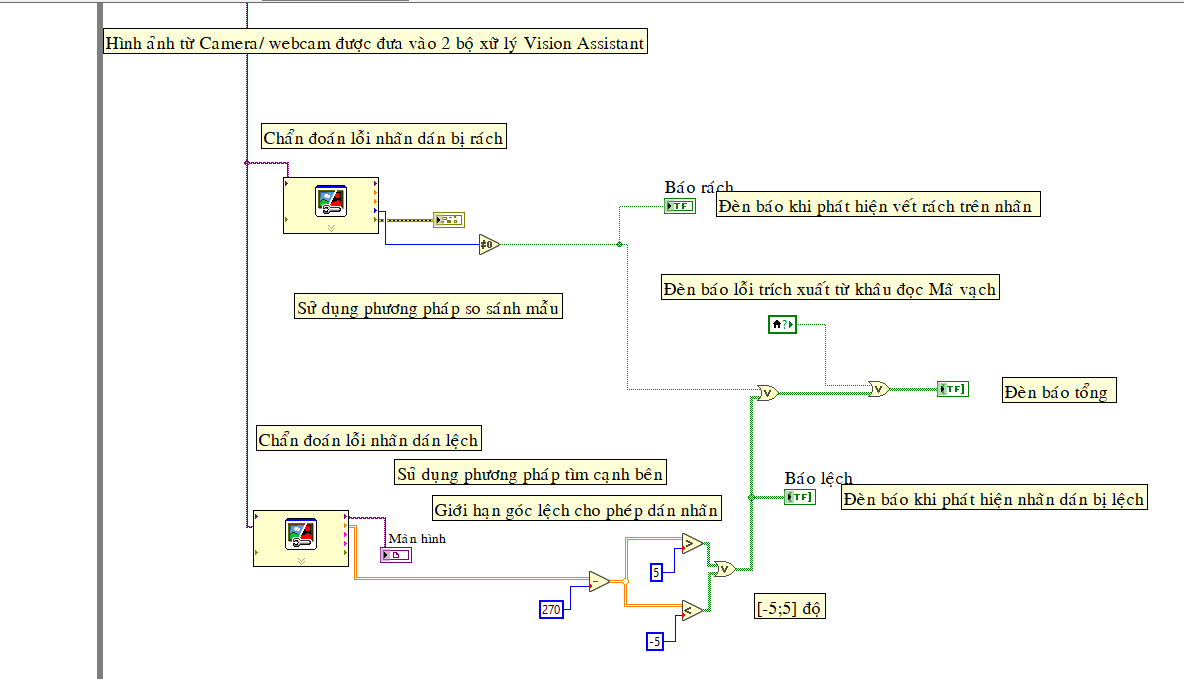
Khi đã có thông tin Data của mã vạch thì chương trình sẽ đưa giá trị đó vào khối so sánh để so sánh với các mã vạch đã được lưu trong thư viện của sản phẩm mà nhà sản xuất đã đăng ký với Quốc tế.

Nếu mã vạch đọc được so sánh trong thư viện mà có thì chương trình sẽ xuất thông tin sản phẩm được đi kèm với mã vạch đó ra màn hình hiển thị cho người dùng dễ dàng quan sát. Ta có thể xem trong hình 3.11

****

Hình 3.11: Truy xuất thư viện

#### 3.2.2.3. Khâu xử lý ảnh Vision Assistant chẩn đoán lỗi dán nhãn

 Hình 3.12: Khối xử lý ảnh Vision chẩn đoán lỗi dán nhãn

Trong chương trình, các khâu xử lý đọc Barcode, chẩn đoán lỗi nhãn dán bị rách, chẩn đoán lỗi dán nhãn lệch đã được tách ra làm 3 khâu riêng biệt để dễ dàng xử lý từng phần một và không để ảnh hưởng của các khâu với nhau.

Trong khâu xử lý ảnh, em sử dụng khối chức năng **Vision Assittant** cho hai lỗi dán lệch, nhãn dán bị rách. Khối chức năng này là khối mà được dùng phổ biến nhất trong bài toán xử lý ảnh vì nó chứa gần hết các khối chức năng trong Module hỗ trợ là NI vision development Module và NI vision Acquisition Software. Giúp cho việc thiết lập cài đặt về thông số cũng như đưa ra các giải thuật một cách khách quan vì có thể quan sát trực tiếp kết quả mà mình trỉnh sửa. Do đó tránh được các lỗi không mong muốn trong quá trình xây dựng chương trình. Ta có thể xem ở hình 3.12.

#### 3.2.2.4 Xây dựng giải thuật cho bài toán chẩn đoán lỗi nhãn dán bị rách

Để xây dựng giải thuật này đầu tiên chúng ta sẽ phải xác định độ phân giải mà Webcam có thể thu được. Như em đã đề cập, ánh sáng là yếu tố cực kỳ quan trọng trong việc thu nhận hình ảnh, ánh sáng tốt thì sẽ thu được hình ảnh rõ ràng và làm cho việc xử lý hình ảnh trở lên tốt hơn, chính xác hơn.



Hình 3.13: Thiết lập thuật toán chẩn đoán lỗi rách nhãn dán

Khi ta thu được hình ảnh rõ nét rồi thì ta bắt đầu tiến hành xây dựng giải thuật. Các giải thuật đều được dựa trên hình ảnh thu được tức là hình ảnh thu được đã được trỉnh về mặt ánh sáng cũng như độ cân đôi giữa vật và Webcam.

Hình ảnh thu được từ khâu xây dựng kết nối Webcam ở mục 3.2.2.1 sẽ được đưa vào khối chức năng Vision Assistant. Trong khối chức năng này, hình ảnh sẽ được đưa qua các khối được xây dựng trong hình 3.13.

Đầu tiên, hình ảnh sẽ được đưa vào khối **Color Plane Extraction** để triết xuất mặt phẳng màu (ở đây em chọn là RGB – Red plane). Lúc này hình ảnh thu được là hình ảnh mà các điểm đỏ được biểu diễn thành các điểm trắng và các màu khác được biểu diễn thành các điểm màu đen hoặc xám tối. Sau đó, ta tiến hành lấy hình mẫu chuẩn trong khối Pattern Matching để hỗ trợ cho khối Map Defects – tìm khuyết điểm. Trong giải thuật này, em đang áp dụng kỹ thuật so sánh mẫu cùng với kỹ thuật trừ điểm ảnh để tìm ra khuyết điểm trên nhãn dán và được thực hiện trong 2 khối **Pattern Matching** và **Map Defects**. Khi ta tìm được khuyết điểm rồi thì ta chuyển hình ảnh ở khối Map Defects thành hình ảnh 8 bit (trước đó hình ảnh là hình ảnh 32 bit) sử dụng khối **Threshold** để tạo vùng cho khuyết điểm tức là làm nổi khuyết điểm lên, đưa khuyết điểm thành màu đỏ và các vùng khác thành màu đen đây là kỹ thuật Trừ ảnh. Sau đó, ta sử dụng khối **Particle filter** để xác định vị trí và giá trị vùng mà đã được tạo trước đó. Cuối cùng là được giá trị thu được chuyển vào khối **Particle Analysis** để đếm xem có bao nhiêu lỗi đã được phát hiện.

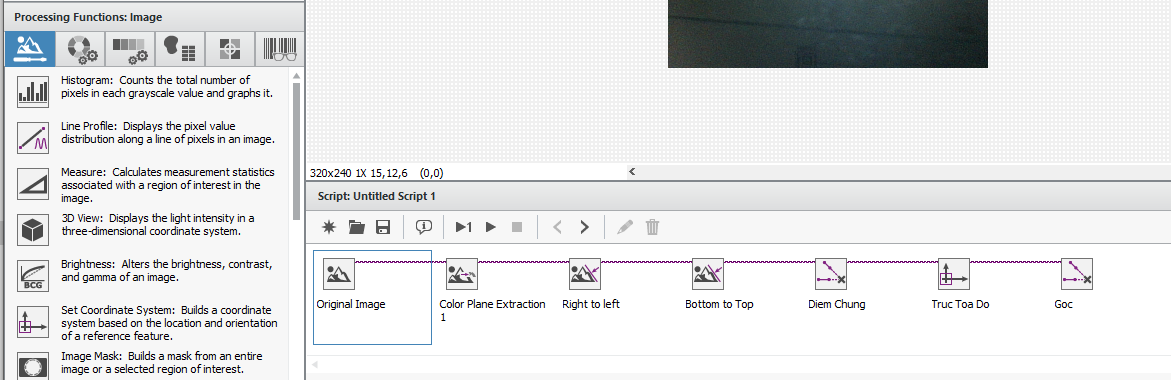
#### 3.2.2.5. Xây dựng giải thuật cho bài toán chẩn đoán lỗi dán nhãn bị lệch

Trong khâu xây dựng giải thuật này em cũng dựa trên hướng làm ở mục 3.2.2.4. Đầu tiên, ta cũng chuyển hình ảnh từ ảnh màu sang ảnh RGB – Red plane để làm cho việc phân vùng màu đen và màu trắng được rõ nét hơn. Giúp cho việc sử dụng kỹ thuật Edge Detection (Phát hiện cạnh) một cách chuẩn xác hơn.

Ở đây, khi hình ảnh đã được tiền xử lý trước đó sẽ được đưa vào 2 khối **Right to Left** và **Bottom to Top** để tìm các cạnh của nhãn dán. Trong đó, **Right to Left** dùng để tìm cạnh bên từ phải qua trái, **Bottom to Top** dùng để tìm cạnh ngang từ dưới lên trên. Để xác định được các cạnh như em đã đề cập ở trên đó là phân vùng màu trắng và đen tức là màu sản phẩm của em là màu đỏ, nhãn dán là màu đen nên em trích xuất màu đỏ ra làm cho màu sản phẩm được đưa vào và hiện nên màn hình là màu trắng, còn màu nhãn dán là màu đen. Dựa trên trên điều đó và em kết hợp với điều kiện mà 2 khối chức năng đưa ra đó là sự chuyển đổi giá trị Pixel một cách đột ngột (ở đây là giá trị điểm ảnh trắng và giá trị điểm ảnh đen có giá trị cách biệt rất lớn). Lúc này giữa trên các biên được xây dựng ở những điểm đột ngột thay đổi đó sẽ tạo ra một đường thẳng và đó là cạnh mình cần xác định.

Để xác định góc lệch thì ta phải thiết lập một trục tọa độ 2 chiều Oxy, gốc tọa độ phải được đặt ở điểm giao nhau của 2 cạnh đã tìm được. Sau đó, ta cố định một cạnh (ở đây cố định cạnh dưới vì sản phẩm của em được đặt dọc theo chiều thu của Webcam) rồi cạnh còn lại để tự do tức có thể quay nhưng được cố định gốc tọa độ. Khi đó, ta có thể đo được góc lệch một cách dễ dàng.

Tuy nhiên, cách làm này mang có hạn chế là góc quay không được quá phạm vi khung xác định cạnh được hiển thị trên màn hình quan sát.

****

Hình 3.14: Thiết lập thuật toán chẩn đoán lỗi lệch nhãn dán

## 3.3. Mô hình thực nghiệm

Mô hình thực nghiệm của em được trên mô hình chung là mô hình cơ bản nhất trong lĩnh vực thị giác máy tính. Đặc biệt là mô hình được sử dụng trong các khâu kiểm tra sản phẩm trong các ngành sản xuất.

### 3.3.1. Mô hình thực tế

Trong mô hình này, em sử dụng 1 chiếc máy tính và một Webcam được kết nối trực tiếp với máy tính qua dây USB. Để hỗ trợ cho việc thu nhận hình ảnh chính xác thì em đã thiết kế một hòm gỗ hình chữ nhật và để trống 3 đầu (để chuyền sản phẩm bên ngoài vào) được tính toán dựa trên khả năng thu hình ảnh của Webcam với độ rộng và độ xa mà Webcam có thể nhận diện sản phẩm. Và cùng với đó không thể thiếu đó là hệ thống chiếu sáng gồm có: 2 đèn Led (được sử dụng trong tivi) cho ánh sáng trắng, một triết áp 2k ôm để giúp cho việc chỉnh ảnh sáng một cách ổn định và dễ dàng hơn. Và sản phẩm mẫu là lọ tương ớt Chin – su của hảng Masan Consumer được công ty TNHH Một thành viên Công nghệ Masan sản xuất gồm có 3 loại : 2 chai đạt chuẩn và có mã vạch khác nhau, 1 chai có nhãn dán bị rách. Ta có thể xem ở 2 hình dưới đây.



Hình 3.15: Mô hình thực tế

### 3.3.2. Giao diện người dùng trên Front Panel

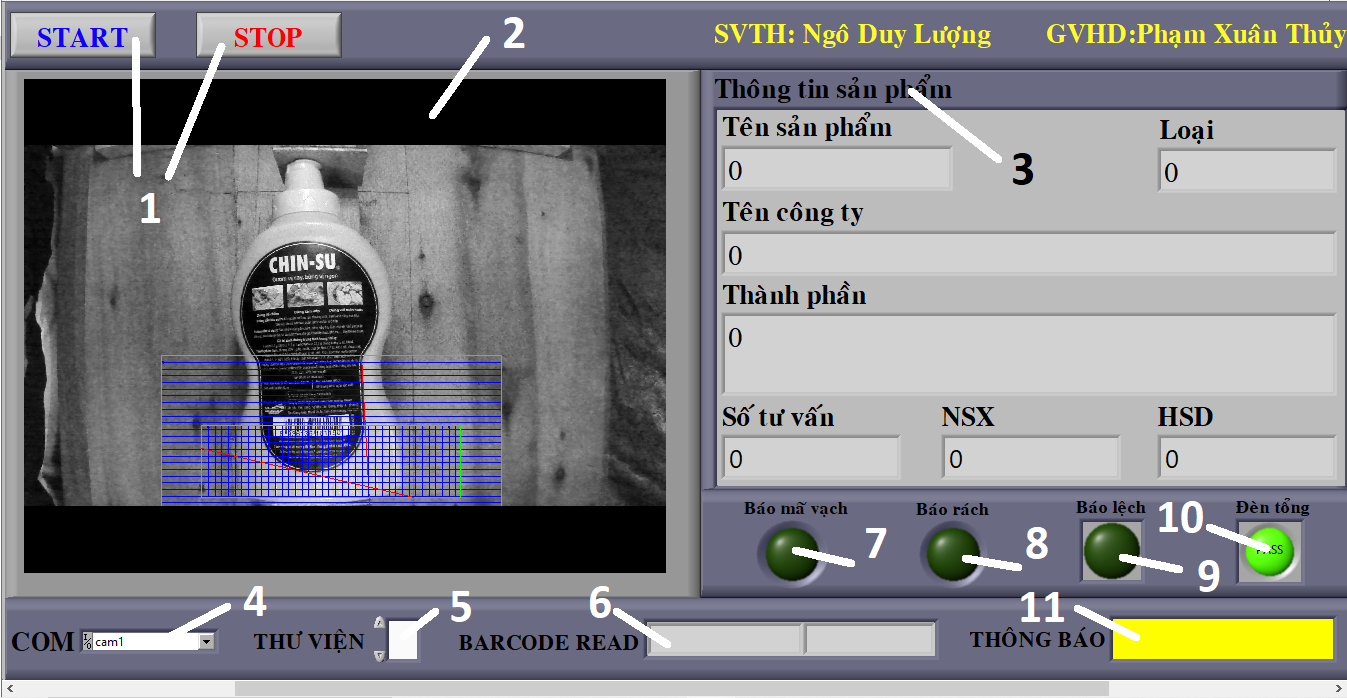
Giao diện người dùng là nơi mà người dùng thực hiện các thiết lập cài đặt để điều khiển cũng như giám sát quá trình mà hệ thống đang hoạt động. Việc tạo nên một giao diện cần phải đủ các tiêu chí như là dễ nhìn, thao tác ở trên đó một cách dễ dàng, và có chứa đầy đủ các thông tin cần thiết. Để đáp ứng được các yêu cầu đó em đã xây dựng một giao diện đơn giản và dễ dàng quan sát và sử dụng được thể hiện dưới hình 3.17.



Hình 3.16: Hệ thống ánh sáng

Trong đó:

1. Nút Stop/Start để dừng/chạy chương trình.
2. Màn hình hiển thị Camera chiếu lên.
3. Bảng hiển thị thông tin sản phẩm.
4. Thiết lập cổng COM được kết với Camera.
5. Chọn thư viện Barcode.
6. Bảng hiển thị giá trị Barcode đọc được từ camera.
7. Đèn báo mã vạch sáng khi Barcode hợp lệ với 1 trong 4 loại Barcode (39,93,128,EAN13) đã thiết lập trong chương trình.

 Hình 3.17: Giao diện người dùng trên Front Panel

1. Đèn báo Rách sáng khi phát hiện ra nhãn dán bị rách.
2. Đèn báo Lệch sáng khi phát hiện ra nhãn dán dán lệch.
3. Đèn tổng sáng khi phát hiện nhãn dán bị lỗi trong 3 lỗi (Dán lệch, Dán bị rách, Mã vạch không đúng).
4. Thanh thông báo Barcode có hợp lệ hay không hợp lệ.

### 3.3.3. Một số kết quả thực nghiệm

a) Trường hợp 1: Dán nhãn sản phẩm đạt yêu cầu

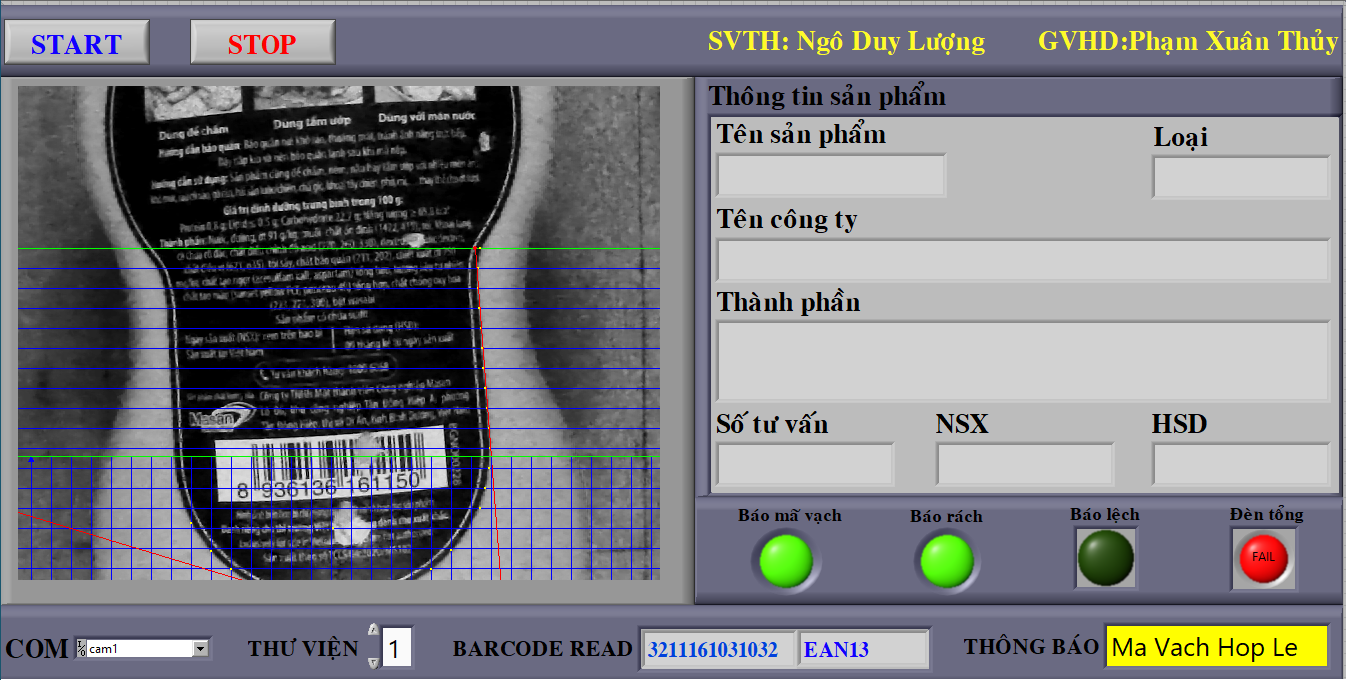
Nhãn dán đặt chuẩn là nhãn dán mà khi đưa qua máy quét hình ảnh sẽ được máy thông báo trên giao diện người dùng. Cụ thể như sau: Trên màn hình quan sát ta thấy dán nhãn đạt chuẩn là khi “*Đèn tổng*” bật sáng và hiện chữ “*PASS*”. Lúc này có thông báo “*Ma Vach Hop Le*” ở ô màu vàng dưới cùng góc phải màn hình, Thông tin sản phẩm được hiện lên trong bảng “Thông tin sản phẩm” bên phải màn hình. Ở dưới màn hình sẽ hiển thị giá trị mã vạch và loại mã vạch mà hệ đọc được. Màn hình hiển thị cho ta thấy sản phẩm lọ tương ớt Chin – su có xuất hiện 2 đường màu đỏ bám vào nhãn dán của sản phẩm – đó là 2 cạnh mà em sử dụng trong giải thuật được nêu ở mục 3.2.2.4



Hình 3.18: Giao diện thông báo sản phẩm đạt chuẩn

Ở đây chúng ta có thể thấy thông tin Data của mã vạch không được hiển thị vì để tránh cho việc làm màn hình hiển thị trở lên dối và khó nhìn nên em đã bỏ phần hiển thị đó và chỉ để xuất hiện các “*cạnh*” để quan sát được nhãn dán lệch như thế nào.

b) Trường hợp 2: Dán nhãn không đạt yêu cầu – nhãn dán bị rách (lỗi)



Hình 3.19: Giao diện thông báo nhãn dán sản phẩm bị rách

Dán nhãn bị rách là nhãn dán khi đi qua máy quét ảnh (Webcam) sẽ được thông báo trên giao diện người dùng như hình 3.19 trên. Cụ thể như sau: đèn “Báo rách” sẽ sáng lên đồng thời “*Đèn tổng*” sẽ chuyển sang màu đỏ mà hiển thị chữ “*FAIL*”. Nếu phần mã vạch không bị rách hoặc mờ thì vẫn sẽ đọc được Barcode trên sản phẩm và hiển thị thông tin như sản phẩm đạt chuẩn. Nếu sai Barcode hoặc bị rách thì máy sẽ không hiển thị thông tin sản phẩm và thanh thông báo sẽ báo “*Ma Vach Khong Hop Le*” đồng thời đèn “*Báo mã vạch*” sẽ không sáng.

c)Trường hợp 3: Dán nhãn sản phẩm không đạt yêu cầu – dán nhãn bị lệch (lỗi)



Hình 3.20: Giao diện thông báo nhãn dán sản phẩm bị dán lệch

Dán nhãn bị lệch sẽ được thông báo hiển thị như ở phần dán nhãn bị rách. Tức là đèn “ *Báo lệch* ” sẽ sáng vào “*Đèn tổng*” hiển thị “*FAIL*”.

**Kết luận chương 3**

Như vậy, trong chương 3, em đã hiện thực hóa thành công một mô hình ứng dụng thị giác máy tính cho bài toán chẩn đoán lỗi dán nhãn sản phẩm. Qua thực nghiệm, có thể kết luận rằng giải pháp thị giác máy tính là một giải pháp rất hữu ích, giúp chúng ta có thể phát triển ở nhiều lĩnh vực khác ngoài lĩnh vực sản xuất.

Tuy nhiên, trong mô hình hệ thống được xây dựng, còn nhiều vấn đề cần bổ xung và sửa chữa. Ví dụ như chưa có khả năng xử lý các bài toán liên quan đến việc dán nhãn sản phẩm vì em mới chỉ xử lý được 2 lỗi chính đó là nhãn dán bị lệch và nhãn dán bị rách. Nếu được khắc phục, nâng cấp, thì chất lượng chẩn đoán có thể được nâng cao hơn.

**KẾT LUẬN CHUNG**

* **Kết quả đạt được:**

Đã thiết kế và chế tạo mô hình hệ thống chẩn đoán lỗi nhãn dán sản phẩm dựa trên một webcam và phần mềm LabVIEW đã đạt được các tiêu chí đề ra bao gồm:

* Giao diện chương trình thân thiện dễ nhìn và sử dụng.
* Độ chính xác với các tiêu chí như sau: Đọc thông tin Barcode 99%; chẩn đoán lỗi lệch nhãn 95%; chẩn đoán lỗi bị rách nhãn 75%.
* **Hạn chế và hướng phát triển**

- Mô hình chỉ dừng lại ở phân tích so sánh mẫu và sử dụng phát hiện cạnh lên chương trình chưa được linh hoạt. Chưa cho phép người dùng sử dụng mở rộng ứng dụng cho nhiều loại Label và hộp in khác nhau, cũng như việc phát hiện ra các lỗi khác ngoài các lỗi mà đã được đề cập.

- Việc ứng dụng thị giác máy tính tương tự như mô hình trên đây giúp cho việc mở ra hướng đi trong việc loại bỏ việc kiểm tra bằng thủ công, hạn chế sai sót trong quá trình sản xuất. Ví dụ có thể ứng dụng trong các lĩnh vực tương tự như kiểm tra kích thước trên bao bì sản phẩm trong nhà máy đóng chai, kiểm tra độ đồng tâm độ tương đồng so với mẫu quy chuẩn trong lĩnh vực cơ khí chế tạo như ổ bi, gia công cơ khí….

- Trong đề tài, em mới chỉ giải quyết 3 vấn đề cơ bản: đọc mã vạch, lỗi dán nhãn bị lệch, nhãn dán bị rách cho bài toán chẩn đoán lỗi dán nhãn sản phẩm. Tuy nhiên, trong việc dán nhãn sản phẩm còn có nhiều lỗi khác, ví dụ như: nhãn dán bị gấp, bị in mờ, nhãn dán chưa được dán vào sản phẩm… Trong thời gian tới, em sẽ phát triển, bổ sung giải pháp để hệ thống nhận ra các lỗi trên và để kết quả của đề tài có thể tiến tới gần hơn vảo trong ứng dụng thực tế.

Một lần nữa, em xin gửi lời cảm ơn đến các Thầy cô giáo, và đặc biệt là thầy **Phạm Xuân Thủy**, người đã tận tình hướng dẫn để em hoàn thành đề tài này.

# TÀI LIỆU THAM KHẢO

**Tiếng Việt**

1. Nguyễn Bá hải (2013), Lập Trình LabVIEW- NXB đại học quốc gia TpHCM

**Tiếng Anh**

* 1. Practical Guide to Machine Vision Software, An Introduction with LabVIEW, Kye-Si Kwon and Steven Ready
  2. Paper\_Identification of Defect in Labels on Cigarette Packets using LabVIEW
  3. Machine Vision using LabVIEW for Label Inspection, Puranam Revanth Kumar1 , T Ananthan2

**Internet**

1. http://www.ni.com/en-vn.html