**ĐẠI HỌC ĐÀ NẴNG**



**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CƠ KHÍ**

** **

**-----🙞🙜🕮🙞🙜-----**

**ĐỒ ÁN KỸ THUẬT VI ĐIỀU KHIỂN**

***ĐỀ TÀI:***

**ỨNG DỤNG VI ĐIỀU KHIỂN PIC**

**VÀ XỬ LÝ ẢNH VÀO PHÂN LOẠI**

**HÌNH DẠNG SẢN PHẨM**

****

GVHD: TS. VÕ NHƯ THÀNH

SVTH: HOÀNG VĂN DUY

HOÀNG ANH TUẤN

LỚP : 15CDT1 NHÓM: 05A

*Đà Nẵng, tháng 12 năm 2018*

**LỜI NÓI ĐẦU**

Công nghệ xử lý ảnh ngày càng được ứng dụng rộng rãi trong cuộc sống. Ngoài các ứng dụng truyền thống như phục hồi, nâng cao chất lượng ảnh, các ứng dụng nhận dạng, an ninh, điều khiển ngày càng phổ biến, trong hệ thống sản xuất tự động. Công nghệ xử lý ảnh và nhận dạng là công nghệ khá phức tạp. Ảnh thu nhận được từ camera luôn bị nhiễu. Việc xử lý khử nhiễu và nhận dạng mục tiêu trở nên khó khăn và tiêu tốn nhiều tài nguyên.

Phần mềm MATLAB đã trở thành công cụ không thể thiếu của các cán bộ nghiên cứu giảng dạy, sinh viên đại học, cao học và nghiên cứu sinh thuộc các ngành khoa học kỹ thuật nước ta. Điều này có được là do MATLAB cung cấp một công cụ tính toán và lập trình bậc cao dễ sử dụng hiệu quả và thân thiện với người dùng. MATLAB cung cấp cho người sử dụng các phương thức để thực hiện các mô phỏng trên máy tính, ngoài ra MATLAB còn có thể giao tiếp với Vi Xử Lý, PLC,…cùng với các thiết bị phần cứng để thực hiện các ứng dụng vào thực tế. Vì vậy trong đồ án môn học, chúng em đã tìm hiểu và thực hiện đề tài “Ứng dụng vi điều khiển PIC và xử lý ảnh vào phân loại hình dạng sản phẩm”.

Trong quá trình làm Đồ án Vi điều khiển, ngoài sự nổ lực của bản thân là sự giúp đỡ vô cùng nhiệt tình của thầy giáo **TS.Võ Như Thành** để chúng em hoàn thành tốt học phần này. Tuy nhiên do kiến thức và khả năng còn có hạn nên khó tránh khỏi những thiếu sót. Chúng em rất mong nhận được đóng góp ý kiến của các thầy và các bạn sinh viên để Đồ án của chúng em được hoàn chỉnh hơn. Xin chân thành cảm ơn.

Đà Nẵng, tháng 12 năm 2018

**MỤCLỤC**

**LỜI NÓI ĐẦU…………………………………………………………………….1**

**MỤC LỤC………………………………………………………………………….2**

**CHƯƠNG 1: PHÂN TÍCH VÀ LỰA CHỌN PHƯƠNG ÁN**

**THIẾT KẾ………………………………………………………….5**

* 1. Giới thiệu chung………………………………………………………………5

1.2 Cấu tạo………………………………………………………………………..6

1.3 Phân loại………………………………………………………………………6

1.4 Phân tích và lựa chọn phương án thiết kế…………………………………….7

1.4.1 Phân tích………………………………………………………………...7

1.4.2 Chọn phương án………………………………………………………...9

1.5 Các chi tiết, linh kiện sử dụng trong hệ thống………………………………..9

1.5.1 Động cơ DC giảm tốc…………………………………………………...9

1.5.2 Động cơ DC servo SG90……………………………………………....10

**CHƯƠNG 2: GIỚI THIỆU VỀ XỬ LÝ ẢNH VÀ GIAO DIỆN GUIDE**

**TRONG MATLAB ………………………………………………11**

2.1 Tổng quan về xử lý ảnh……………………………………………………11

2.2 Các khái niệm và hàm trong xử lý ảnh…………………………………….11

2.2.1 Ảnh và điểm ảnh………………………………………………………11

2.2.2 Độ phân giải ảnh………………………………………………………12

2.2.3 Khử nhiễu……………………………………………………………..12

2.2.4 Đặc điểm biên và đường biên…………………………………………12

2.2.5 Mức xám của ảnh………………………………………………………12

2.2.6 Ảnh chỉ số………………………………………………………………13

2.2.7 Ảnh nhị phân………………………………………………………..…14

2.2.8 Ảnh màu…………………………………………………………….…14

2.3 Các hàm xử lý ảnh thông dụng………………………………………………15

2.4 Bắt ảnh với Matlab……………………………………………………….…18

2.5 Giao diện GUIDE trong matlab……………………………………….……..21

**CHƯƠNG 3: GIỚI THIỆU VỀ VI ĐIỀU KHIỂN PIC16F877A**

3.1 Giới thiệu chung………………………………………………………….….24

3.1.1 PIC là gì…………………………………………………………..…….24

3.1.2 Đặc điểm của PIC…………………………………………………..…..24

3.1.3 Các loại chip PIC………………………………………………….…….24

3.1.4 Trình biên dịch CCS……………………………………………….……25

3.2 Tổng quan về PIC16F877A………………………………………….………25

3.2.1 Giới thiệu………………………………………………………….…….25

3.2.2 Sơ đồ chân và chức năng chân PIC16F877A…………………….……26

3.2.3 Các đặc tính ngoại vi…………………………………………….……..30

3.2.4 Giao tiếp USART……………………………………………….………31

3.2.5 Giới thiệu về giao tiếp UART – USB TTL……………………….…….36

**CHƯƠNG 4: ĐIỀU KHIỂN HỆ THỐNG BĂNG TẢI PHÂN LOẠI HÌNH**

**DẠNG SẢN PHẨM BẰNG XỬ LÝ ẢNH………………………38**

4.1 Sơ đồ nguyên lý và nguyên lý hoạt động của mạch điều khiển………..……38

4.1.1 Sơ đồ nguyên lý………………………………………………..………38

4.1.2 Nguyên lý hoạt động……………………………………………………38

4.1.3 Lưu đồ thuật toán………………………………………………….……40

4.1.4 Code PIC………………………………………………………………..40

4.2 Quá trình xử lý ảnh và code Matlab…………………………………………42

4.2.1 Ý tưởng………………………………………………………….………42

4.2.2 Lưu đồ thuật toán……………………………………………………….44

4.2.3 Code chương trình………………………………………………...……45

4.3 Mô phỏng thực tế……………………………………………………………49

**CHƯƠNG 1: PHÂN TÍCH VÀ LỰA CHỌN PHƯƠNG ÁN THIẾT KẾ HỆ THỐNG**

**1.1 Giới thiệu chung về băng tải**

- Các băng tải thường dùng để di chuyển các vật liệu đơn chiếc và vật liệu rời theo phương ngang, phương đứng, phương xoắn. Trong các dây chuyền sản xuất, các thiết bị này được sử dụng rộng rãi như những phương tiện vận chuyển các linh kiện nhẹ, trong các xưởng kim loại được dùng để vận chuyển quặng, than đá, các loại xí lò; trên các trạm thủy điện thì dùng để chuyển nhiên liệu; trên các kho bãi thì dùng vận chuyển các loại hàng bao kiện vật liệu hạt hoặc một số sản phẩm khác; trên các công trường  thì dùng để vận chuyển vật liệu xây dựng; trong ngành lâm nghiệp và khai thác gỗ thì vận chuyển gỗ, vỏ bào; trong một số ngành công nghiệp nhẹ, công nghiệp thực phẩm, hóa chất và một số ngành công nghiệp khác thì dùng để vận chuyển sản phẩm hoàn thành và chưa hoàn thành ở các giai đoạn, các phân xưởng, đồng thời cũng như loại bỏ các sản phẩm không dùng được.

- Khác với các thiết bị vận chuyển khác, băng tải có chiều dài vận chuyển lớn, năng suất lớn, kết cấu nhỏ, đơn giản, làm việc tin cậy và sử dụng thuận tiện.



*Hình 1.1: Hệ thống băng tải*

- Ngày nay người ta sử dụng băng tải có độ bền cao, chiều rộng tới 3m tốc độ vận chuyển có thể đạt tới 4 km/h và hơn nữa năng suất của băng tải có thể đạt vài nghìn tấn trong một giờ. Trên thực tế chiều dài mang tải không giới hạn và có thể áp dụng hệ thống gồm nhiều giai đoạn liên kết. Những hệ thống này được sử dụng rộng rãi trong ngành khai thác mỏ quặng, cũng như ngành xây dựng. Ở đó băng tải có khả năng cạnh tranh lớn với đường chuyển cáp treo, thậm chí cả đối với vận chuyển bằng ô tô, đường sắt.



*Hình 1.2: Băng tải*

**1.2 Cấu tạo**

- Một động cơ giảm tốc trục vít, bộ truyền xích,.. và bộ điều khiển kiểm soát tốc độ.

- Bộ con lăn chủ động và bị động.

- Hệ thống khung đỡ con lăn

- Hệ thống dây băng hoặc con lăn.  
**1.3 Phân loại**

\* Theo phương chuyển động:

- Theo phương ngang

- Theo phương nghiêng

- Theo phương đứng

- Theo phương xoắn

\* Theo kết cấu:

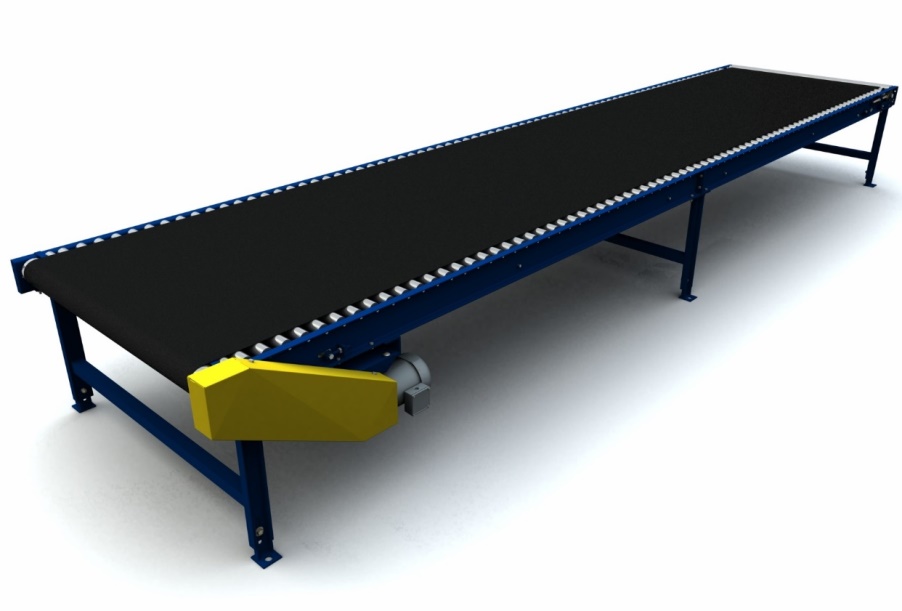
- Loại cố định

- Loại di động

**1.4 Phân tích và lựa chọn phương án.**

**1.4.1 Phân tích**

**a) Băng tải thẳng**

****

*Hình 1.3: Băng tải thẳng*

- [Băng tải thẳng](http://conlan.vn/Bang-tai-Bang-chuyen-PVC/bang-tai-pvc-thang-64.html) thường được sử dụng trong các nhà máy sản xuất, chế biến và lắp ráp linh kiện ở các khu công nghiệp mỏ khai khoáng để đáp ứng các nhu cầu sản xuất công nghiệp và vận chuyển vật liệu nặng.

**\* Ưu điểm nổi bật của băng tải thẳng:**

- Băng tải cấu tạo đơn giản và có độ bền cao  
 - Không gây tiếng ồn  
 - Vận chuyển hàng hóa không làm rơi rớt, trơn trượt quá trinh vận chuyển với nhiều vận tốc.  
 - Dễ dàng kiểm soát và điều chỉnh tốc độ, vận hành nên thuận lợi cho việc sử dụng  
 - Tuổi thọ cao

**b) Băng tải cong**

**-** [Băng tải con lăn](http://bangtaithanhcong.com/bang-tai-con-lan/)**góc cong** là loại băng tải chuyển hướng di chuyển sản phẩm với góc độ 450; 600, 900, 1800. Được chia ra làm 2 loại có động cơ và không có động cơ.

- Băng tải góc cong được sử dụng  để vận chuyển tất cả các loại mặt hàng khác nhau như hộp, túi xách,quần áo . Ngoài ra băng tải đáp ứng khả năng chống dầu, chống ăn mòn, vật liệu chống tĩnh điện như yêu cầu đặc biệt vận chuyển.

- Băng tải có khả năng phối hợp lớn và hiệu quả, kết nối thuận tiện với các băng tải khác dễ dàng.

**-** [Băng tải](http://bangtaithanhcong.com/bang-tai-bang-chuyen-chat-luong-cao/)**góc cong** được đặt ở vị trí chuyển hướng di chuyển của băng tải để tiết kiệm không gian.



*Hình 1.4: Băng tải góc cong*

**1.4.2 Chọn phương án**

\* Sau khi nghiên cứu thì nhóm chọn loại băng tải thẳng để phù hợp với đồ án.

- Cơ cấu phân loại sản phẩm thì có 2 phương án:

+ Phương án 1: Dùng 1 động cơ servo

+ Phương án 2: Dùng 2 động cơ servo

=> Để giảm bớt chi phí và đơn giản hệ thống thì nhóm chọn phương án 1 với cơ

cấu phân loại dùng 1 động cơ servo để điều khiển.

- Cơ cấu quay phân loại được gắn động cơ servo để quay các góc khác nhau đưa sản phẩm đến các hộp đựng tương ứng.

**1.5 Các chi tiết, linh kiện sử dụng trong hệ thống**

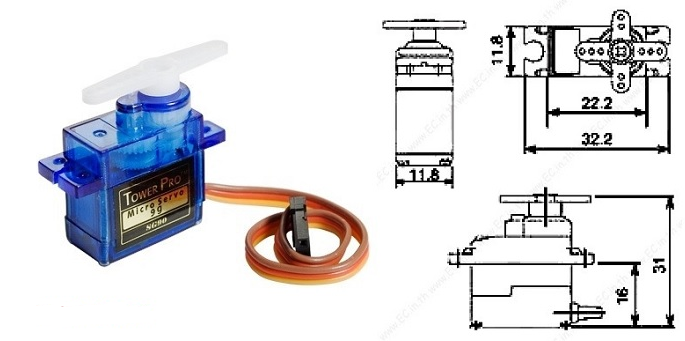
**1.5.1 Động cơ DC giảm tốc.**



*Hình 1.5: Động cơ DC giảm tốc*

**-** Động cơ DC giảm tốc dùng để quay băng tải trong mô hình đồ án nên nhóm em chọn loại quay với tốc độ chậm để giảm tốc độ băng tải nhằm mục đích giảm nhiễu và mờ ảnh được chụp từ băng tải.

**1.5.2 Động cơ DC servo SG90**



*Hình 1.6: Động cơ servo*

- Thông số kỹ thuật:

+ Khối lượng : 9g

+ Kích thước: 22.2x11.8.32 mm

+ Momen xoắn: 1.8kg/cm

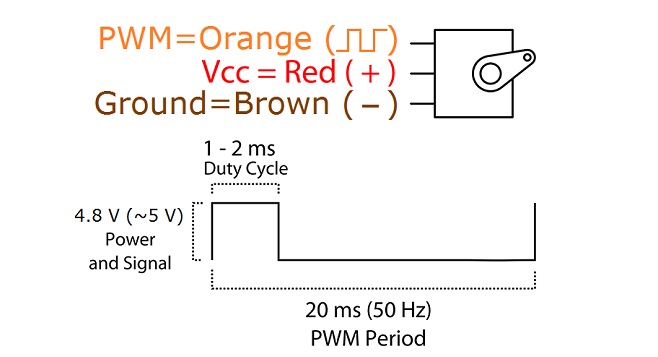
+ Tốc độ hoạt động: 60 độ trong 0.1 giây

+ Điện áp hoạt động: 4.8V(~5V)

+ Nhiệt độ hoạt động: 0 ºC – 55 ºC

- Cách điều khiển:

Kết nối dây màu đỏ với nguồn 5V, dây màu nâu với mass, dây màu cam nối với chân phát xung của vi điều khiển. Ở chân xung cấp một xung từ 1ms-2ms theo để điều khiển góc quay theo ý muốn.



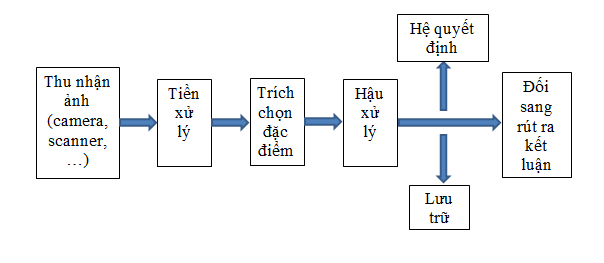
*Hình 1.7: Sơ đồ điều khiển động cơ servo*

**CHƯƠNG 2: GIỚI THIỆU VỀ XỬ LÝ ẢNH VÀ GIAO DIỆN GUIDE TRONG MATLAB**

**2.1 Tổng quan về xử lý ảnh**

- Matlab cung cấp Image processing toolbox, chuyên về xử lý ảnh. Có thể nói Matlab là một công cụ lợi hại giúp cho việc thực hiện các giải thuật xử lý ảnh nhanh chóng và dễ hiểu.

- 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” hoặc lệnh điều khiển cho hệ thống chấp hành.



*Hình 2.1 Các bước xử lý ảnh*

**2.2 Các khái niệm và hàm trong xử lý ảnh**

**2.2.1 Ảnh và điểm ảnh**

- Ảnh được xem như là 1 tập hợp các điểm ảnh.

- Điểm ảnh 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.

**2.2.2 Độ phân giải ảnh**

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

**2.2.3 Khử nhiễu**

\* Có 2 loại nhiễu cơ bản trong quá trình thu nhận ảnh

- Nhiều hệ thống: là nhiễu có quy luật có thể khử bằng các phép biến đổi

- Nhiễu ngẫu nhiên: vết bẩn không rõ nguyên nhân → khắc phục bằng các phép lọc

**2.2.4 Đặc điểm biên và đường biên**

Đặc trưng cho đường biên của đối tượng và do vậy rất hữu ích trong việc trích trọn các thuộc tính bất biến được dùng khi nhận dạng đối tượng. Việc trích chọn hiệu quả các đặc điểm giúp cho việc nhận dạng các đối tượng ảnh chính xác, với tốc độ tính toán cao và dung lượng nhớ lưu trữ giảm xuống.

**2.2.5 Mức xám của ảnh**

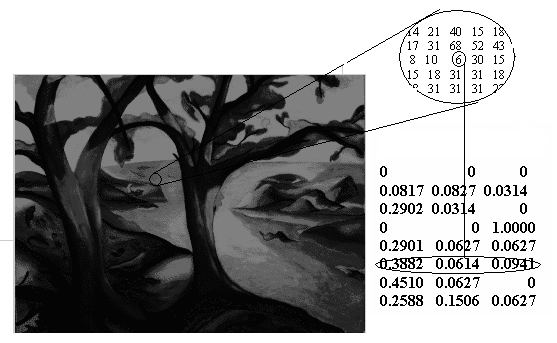
**-** Một điểm ảnh (pixel) có hai đặc trưng cơ bản là vị trí *(x, y)* của điểm ảnh và độ xám của nó.

**-** Định nghĩa: Mức xám của điểm ảnh là cường độ sáng của nó được gánbằng 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: 28=256 mức, tức là từ 0 đến 255).

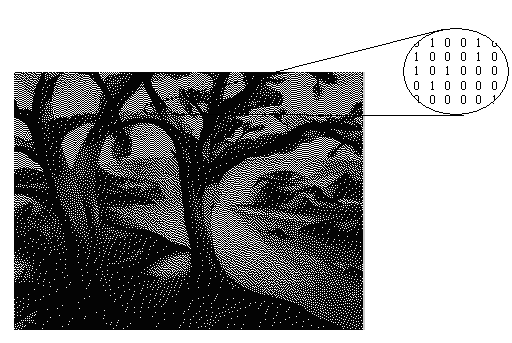
**2.2.6 Ảnh chỉ số**

- Ảnh được biểu diễn bởi hai ma trận, một ma trận dữ liệu ảnh X và một ma trận màu (còn gọi là bản đồ màu). Ma trận có thể là kiểu uint8, uint16 hoặc double. Ma trận màu là một ma trận kích thước m x 3 gồm các thành phần thuộc kiểu double có giá trị trong khoảng [0 1]. Mỗi hàng của ma trận xác định thành phần red, green, blue của một màu trong tổng số m màu được sử dụng trong ảnh. Gía trị của một phần tử trong ma trận dữ liệu ảnh cho biết màu của điểm ảnh đó nằm ở hàng nào trong ma trận màu. Nếu ma trận dữ liệu thuộc kiểu double, giá trị 1 sẽ tương ứng với hàng thứ 1 trong bảng màu, giá trị hai tương ứng với màu ở hàng thứ hai,…Nếu ma trận dữ liệu thuộc kiểu uint8 hoặc uint16, giá trị 0 tương ứng với hàng 1, giá tri 1 tương ứng với hàng 2,…Riêng với kiểu uint16, MATLAB không hỗ trợ đủ các phép toán so với kiểu uint8 nên khi cần xử lý ta cần chuyển sang kiểu dữ liệu uint8 hoặc double bằng các hàm imapprox hoặc im2double.



*Hình 2.2: Ảnh chỉ số*

**2.2.7 Ảnh nhị phân:**ảnh chỉ có 2 mức đen trắng phân biệt. Nói cách khác: mỗi điểm ảnh của ảnh nhị phân chỉ có thể là 0 hoặc 1.



*Hình 2.3: Ảnh chỉ số*

**2.2.8 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: 28\*3=224 ≈ 16,7 triệu màu.



*Hình 2.4: Ảnh màu*

**2.3 Các hàm xử lý ảnh thông dụng**

**\* Hàm imread**

- Đọc một ảnh từ bất kì định dạng nào được trợ giúp trong bất kì chiều sâu bit nào được trợ giúp. Hầu hết các file ảnh sử dụng 8 bit để chứa giá trị của pixel. Khi chúng được đọc vào bộ nhớ, Matlab chứa chúng dưới dạng uint8. Với các file trợ giúp 16 bit dữ liệu, PNG và TIFF, Matlab chứa chúng dưới dạng uint16.

- Chú ý: Với ảnh chỉ số, imread luôn luôn đọc bản đồ màu vào trong một chuỗi thuộc lớp double, thậm chí mảng ảnh tự nó thuộc lớp uint8 hay uint16 - Chẳng hạn, đoạn mã sau sẽ đọc một ảnh RGB vào không gian làm việc của Matlab lưu trong biến RGB.

**\* Hàm imwrite**

- Ghi từ ảnh gốc để tạo ra một file ảnh mới

- Ví dụ : imwrite(I,’result.jpg’);

**\* Hàm imfinfo**

- Hàm imfinfo cho phép ta có thể nhận được thông tin về một file ảnh được trợ giúp bởi toolbox.

- Cú pháp: imfinfo(filename,fmt)

- Các thông tin được cung cấp bởi hàm imfinfo là: filename, filemodedate, filesize, format, formatversion, width, height, bitdepth, colortype …

- Thông tin mà ta nhận được phụ thuộc vào kiểu của file nhưng nó luôn bao gồm những thông tin sau:

+ Tên của file ảnh.

+ Định dạng file ảnh.

+ Số version của định dạng file.

+ Ngày sửa đổi file gần nhất.

+ Kích thước file tính theo byte.

+ Chiều rộng ảnh tính theo pixel.

+ Chiều cao ảnh tính theo pixel.

+ Số lượng bít trên một pixel.

+ Kiểu ảnh: RGB, chỉ số …

\* **Hiển thị ảnh**

- Dùng hàm **imview**

+ Để hiển thị một ảnh sử dụng hàm imview, chỉ rõ ảnh mà ta muốn hiển thị. Ta có thể sử dụng imview để hiển thị một ảnh mà đã được nhập vào trong không gian làm việc của Matlab.

+ moonfig = imread('moon.tif');

+ imview(moonfig);

- Dùng hàm **imshow**

+ Để xem ảnh, ta có thể sử dụng hàm imshow thay cho imview. Ta sử dụng imshow để hiển thị một ảnh đã được nhập vào trong không gian làm việc như ví dụ sau:

+ img = imread('anh.tif'); imshow(img);

**\* Hàm chuyển đổi kiểu ảnh**

- Khi chuyển đổi một ảnh từ dạng này sang dạng khác, ảnh kết quả có thể khác ảnh ban đầu.Chẳng hạn, nếu ta chuyển đổi một ảnh màu chỉ số sang một ảnh cường độ, kết quả ta sẽ thu được một ảnh đen trắng.

- Danh sách sau đây sẽ liệt kê các hàm được sử dụng trong việc chuyển đổi kiểu ảnh:

+ dither : Tạo một ảnh nhị phân từ một ảnh cường độ đen trắng bằng cách trộn, tạo một ảnh chỉ số từ một ảnh RGB bằng cách trộn.

+ gray2id : Tạo một ảnh chỉ số từ một ảnh cường độ đen trắng.

+ grayslice : Tạo một ảnh chỉ số từ một ảnh cường độ đen trắng bằng cách đặt ngưỡng.

+ im2bw : Tạo một ảnh nhị phân từ một ảnh cường độ, ảnh chỉ số hay ảnh RGB trên cơ sở của ngưỡng ánh sáng

+ ind2gray : Tạo một ảnh cường độ đen trắng từ một ảnh chỉ số.

+ ind2rgb : Tạo một ảnh RGB từ một ảnh chỉ số.

+ rgb2gray : Tạo một ảnh cường độ đen trắng từ một ảnh RGB.

**\* Quay ảnh**

- Để quay một ảnh, sử dụng hàm imrotate. Hàm này chấp nhận hai tham số chính: + Ảnh cần quay + Góc quay - Góc quay tính theo độ. Nếu ta chỉ ra một giá trị dương, hàm imrotate quay ảnh theo chiều ngược chiều kim đồng hồ. Nếu chỉ ra giá trị âm, hàm quay ảnh theo chiều kim đồng hồ. Ví dụ sau quay một ảnh 35 độ theo chiều ngược chiều kim đồng hồ:

- Ví dụ: J=imrotate(I,35 ) ;

**\* Ngưỡng ảnh**

- Level = graythresh(I);

- Bw=im2bw(I,level); % chuyển sang ảnh nhị phân dựa trên ngưỡng

- Figure,imshow(Bw);

\* **Hàm rgb2gray()**

- Lệnh**rgb2gray()**chuyển đổi hình ảnh RGB thành trắng đen bằng cách loại bỏ các thông tin màu sắc và độ bão hòa nhưng vẫn giữ độ sáng.

-Ví dụ :

img = imread(‘anh.jpg’);

Gray=rgb2gray(img);

Imshow(img);

**\* Hàm imadjust()**

- Hàm **imadjust()** điều chỉnh giá trị cường độ hình ảnh. Câu lệnh này làm

tăng độ tương phản của hình ảnh đầu ra.

- Ví dụ :

img = imread(‘anh.jpg’);

gray=rgb2gray(img);

adj\_img=imadjust(gray,[0.3,0.7],[]); imshow(adj\_img);

**2.4 Bắt ảnh** **với Matlab**

**\* Nhiệm vụ của Toolbox**

- Image Acquisition Toolbox là tập hợp các hàm của MATLAB có nhiệm vụ thu thập hình ảnh số từ các thiết bị thu hình. Cụ thể là:

- Thu thập hình ảnh từ các thiết bị thu hình khác nhau.

- Trình chiếu (preview) luồng dữ liệu video từ thiết bị thu hình.

- Trích hình ảnh từ luồng dữ liệu video.

- Thiết lập các callback cho các sự kiện khác nhau.

- Chuyển dữ liệu hình ảnh vào không gian workspace của MATLAB để xử lý ảnh tiếp theo.

**\* Các bước thu thập hình ảnh cơ bản với Toolbox:**

**Bước 1:** Gắn thiết bị thu hình vào máy tính và cài đặt driver điều khiển

- Gắn thiết bị thu hình vào máy tính.

- Cài đặt driver (trình điều khiển) cho thiết bị.

- Xem thử hình ảnh video hiện trên máy tính thông qua phần mềm của nhà sản xuất.

- Khởi động MATLAB

**Bước 2:** Chỉ định thông tin phần cứng:

- Ta cần chỉ định cho MATLAB biết phần cứng nào dùng để lấy dữ liệu (vì có thể có nhiều camera kết nối với PC).

- Gõ lệnh imaqhwinfo để biết tên các loại thiết bị có thể dùng.

|  |  |
| --- | --- |
| Tên | Loại thiết bị |
| ‘coreco’ | Thiết bị của công ty coreco |
| ‘DCAM’ | Các thiết bị dùng chuẩn IEEE 1394 |
| ‘dt’ | Các thiết bị của Data Translation |
| ‘Matrox’ | Các thiết bị của hãng Matrox |
| ‘winvideo’ | Các thiết bị dùng chuẩn Windows Driver Model (WDM) hoặc Video for Windows (VFW). Bao gồm các loại USB WEBCAM và camera IEEE 1394 |

- Chỉ định thiết bị: Tuỳ vào loại thiết bị (adaptor name), mỗi thiết bị cùng loại gắn vào PC được đánh số ID, thiết bị thứ nhất có deviceID = 1, thiết bị thứ hai có deviceID = 2,…

- Để liệt kê xem loại winvideo có bao nhiêu thiết bị gắn vào máy, ta cũng dùng lệnh imaqhwinfor với đối số là winvideo

imaqhwinfo(‘winvideo’)

ans =

AdaptorDllName:’C:\MATLAB\SupportPackages\R2015a\osgenericvideointerface\toolbox\imaq\supportpackages\genericv…’

AdaptorDllVersion: ‘4.9 (R2015a)’

AdaptorName: ‘winvideo’

DeviceIDs: {[1]}

DeviceInfo: [1x1 struct]

- Có một thiết bị với DeviceID là 1. Ta có thể tìm hiểu thêm thông tin về thiết bị này.

>>imaqhwinfo(‘winvideo’,1)

ans=

DefaultFormat : ‘YUY2\_160x120’

DeviceFileSupported : 0

DeviceName : ‘USB2.0 PC CAMERA’

DeviceID : 1

VideoInputConstructor : ‘videoinput(‘winvideo’, 1)’

VideoDeviceConstructor : ‘imaq.VideoDevice(‘winvideo’, 1)’

SupportedFormats : {‘YUY2\_160x120’ ‘YUY2\_176x144’ ‘YUY2\_320x240’ ‘YUY2\_352x288’ ‘YUY2\_640x480’}Đây là Webcam đã gắn vào máy để làm đề tài này.

**Bước 3:** Tạo đối tượng đại diện cho luồng dữ liệu video

- Để tạo luồng dữ liệu video ta dùng lệnh videoinput.

- Ví dụ:

>>vid = videoinput(‘winvideo’,1);

**Bước 4:** Hiển thị luồng video trên màn hình để xem thử

- Để xem trước luồng dữ liệu video ta dùng lệnh preview.

Ví dụ: preview(vid)

- Màn hình lúc này hiện lên cửa sổ video của đối tượng vid

- Để kết thúc xem ta dùng lệnh stoppreview hoặc để đóng cửa sổ preview ta dùng lệnh closepreview(vid).

**Bước 5:** Lấy một khung ảnh và lưu thành file đồ hoạ.

- Để lấy một ảnh vào xử lý ta dùng lệnh getsnapshot.

- Để lưu ảnh thành file đồ hoạ ta dùng lệnh imwrite.

- Ví dụ:

img = getsnapshot (vid);

imwrite (img, ‘duy.jpg’);

**Bước 6:** Làm sạch bộ nhớ

- Sau khi lấy dữ liệu hình ảnh ta có thể giải phóng bớt bộ nhớ bằng các lệnh sau:

delete(vid)

clear

close(gcf)

**2.5 Giao diện GUIDE trong matlab**

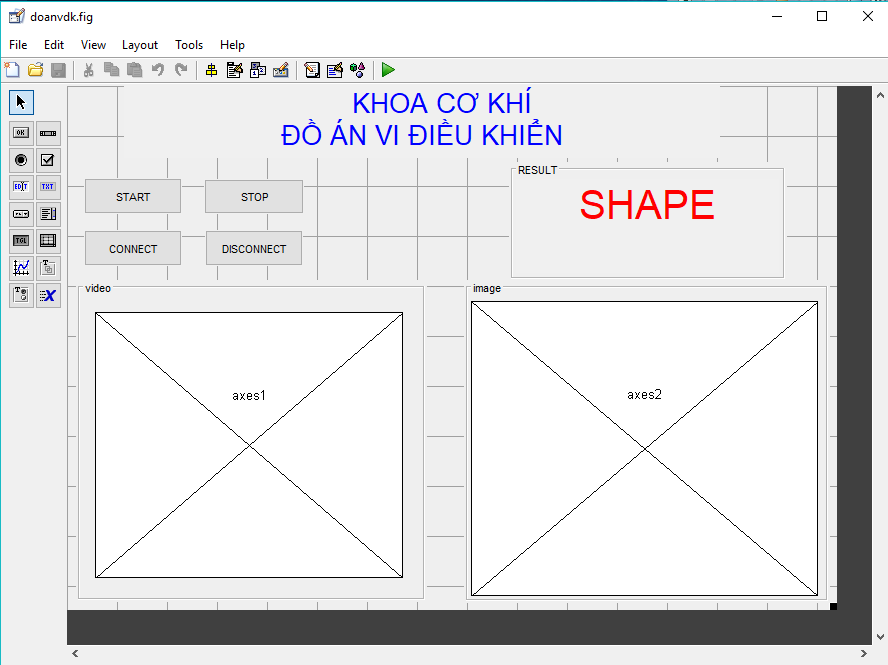
- Guide là giao diện bằng hình ảnh của chương trình. Guide bao gồm các nút nhấn, hộp liệt kê, thanh trượt, menu… chúng cung cấp cho người dùng sử dụng một môi trường làm việc thân thiện để họ tập trung vào các ứng dụng của chương trình hơn là đi tìm hiểu cách thức làm việc của chương trình đó.

- Để mở công cụ tạo Guide : File🡪 New🡪 GUI .

- Khi lưu giao diện vừa tạo Matlab sẽ tạo ra hai file có cùng tên nhưng khác phần mở rộng:

+ File có phần mở rộng .fig chứa nội dung của giao diện.

+ File có phần mở rộng .m chứa những đoạn mã liên quan đến giao diện.



*Hình 2.5: Giao diện Guide*

- Khi thiết kế bất cứ thành phần nào của Guide ta cần thiết lập thuộc tính cho thành phần đó.

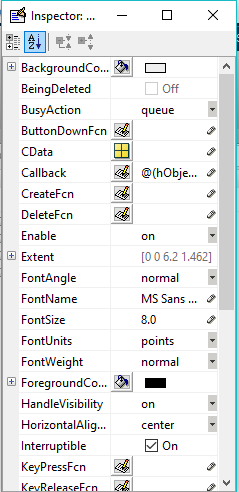
- Để thiết lập các thuộc tính ta có thể chọn mục “ Property Inspector” trên thanh công cụ hoặc right-click vào đối tượng và chọn mục “Inspector Properties”

- Hai thuộc tính quan trọng mà ta cần xác lập là “String Property” và “ Tag Property”.

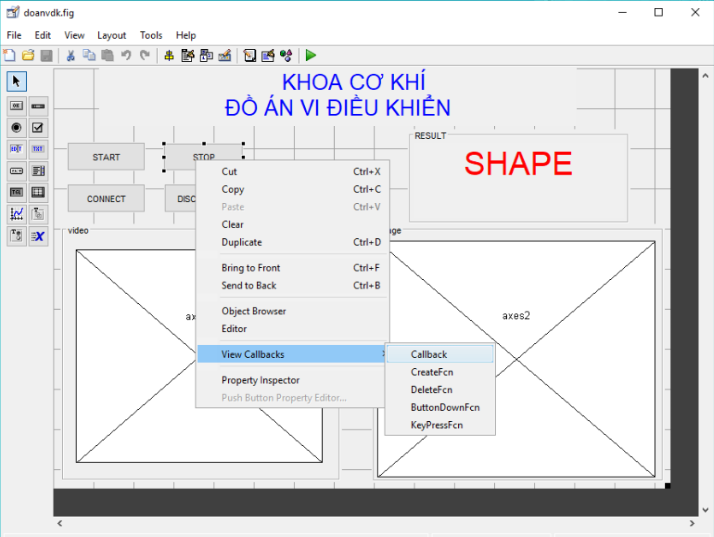
+ String property : dòng ký tự xuất hiện trên đối tượng.

+ Tag property : tên của đối tượng.

- Khi click chuột vào 1 đối tượng, Matlab sẽ gọi hàm tương ứng với đối tượng đó. Tên của hàm chính là tên của đối tượng cộng với“\_Callback”.



*Hình 2.6: Thiết lập giao diện guide*



*Hình 2.7 Lệnh view Callbacks*

- Khi muốn lập trình chức năng cho từng button thì Click phải chọn View Callbacks => Callback

**CHƯƠNG 3: GIỚI THIỆU VỀ VI ĐIỀU KHIỂN**

**PIC16F877A**

**3.1 Giới thiệu chung**

**3.1.1 PIC là gì**

- PIC là viết tắt của “Programable Intelligent Computer”, có thể tạm dịch là “máy tính thông minh khả trình” do hãng General Instrument đặt tên cho vi điều khiển đầu tiên của họ.

- PIC1650 được thiết kế dùng để làm các thiết bị ngoại vi cho điều khiển CP1600. Vi điều khiển này sau đó được nghiên cứu phát triển thêm và từ đó hình thành nên dòng vi điều khiển PIC ngày nay.

- PIC được sản xuất bởi công ty Microchip Technology

**3.1.2 Đặc điểm của PIC**

**-** Có thể tìm mua dễ dàng tại thị trường Việt Nam

- Giá thành không quá cao

- Có đầy đủ tính năng của một VĐK khi hoạt động độc lập

- Số lượng người dùng lớn => Số lượng tài liệu lớn, mã nguồn mở, dễ dàng trao đổi học tập, thảo luận

- Sự hổ trợ của nhà sản xuất về trình biên dịch, các công cụ lập trình, nạp chương trình…

- Các tính năng đa dạng của VĐK luôn được cập nhật

**3.1.3 Các loại chip PIC**

- PIC12xxxx: độ dài lệnh 12 bit

- PIC16xxxx: độ dài lệnh 14 bit

- PIC18xxxx: độ dài lệnh 16bit

- C: PIC có bộ nhớ chương trình là EPROM( chỉ có 16C84 là EEPROM)

- F: PIC có bộ nhớ chương trình là flash

- LF: PIC có bộ nhớ chương trình là flash hoạt động ở điện áp thấp

- LV: tương tự như LF, đây là tín hiệu cũ

- Bên cạnh đó, một số vi điều khiển có ký hiệu xxFxxx thì bộ nhớ chương trình là EEPROM, nếu có thêm chữ A ở cuối thì bộ nhớ chương trình là flash (ví dụ: PIC16F877 là EEPROM, còn PIC16F877A là flash).

**3.1.4 Trình biên dịch CCS**

**-** Kế thừa tất cả đặc điểm của ngôn ngữ C – là ngôn ngữ cơ bản, quen thuộc mà sinh viên đã được đào tạo

- Xây dựng sẵn các hàm phục vụ cho việc sử dụng dễ dàng các khối chức năng đặc biệt của Vi điều khiển PIC như: ADC, PWM, RS232, SPI

- Có khả năng kết hợp với ngôn ngữ hợp ngữ, tạo sự mềm dẻo trong sự phát triển ứng dụng

- Khả năng phát triển, nâng cấp ứng dụng là dễ dàng

- Ngày càng được cập nhật với nhiều tính năng ưu việt và hiệu quả hơn

**3.2 Tổng quan về PIC16F877A**

**3.2.1 Giới thiệu:**

**-** Thuộc họ PIC16Fxxx với tập lệnh gồm 35 lệnh có độ dài 14 bit

- Tất cả các lệnh là 1 chu kỳ máy, ngoại trừ chương trình con là 2 chu kỳ máy

- Một chu kỳ lệnh của vi điều khiển bào gồm 4 xung clock

Sử dụng thạch anh 4MHz => Xung lệnh có tần số 1MHz(1µs)

- Bộ nhớ chương trình flash với dung lượng 8K x 14 bit, với khả năng ghi/xóa 100 000 lần

- Bộ nhớ dữ liệu RAM: 368 type

- Bộ nhớ EEPROM với dung lượng 256 type, với khả năng ghi/xóa 1.000.000 lần. Có thể lưu trữ dữ liệu hơn 40 năm

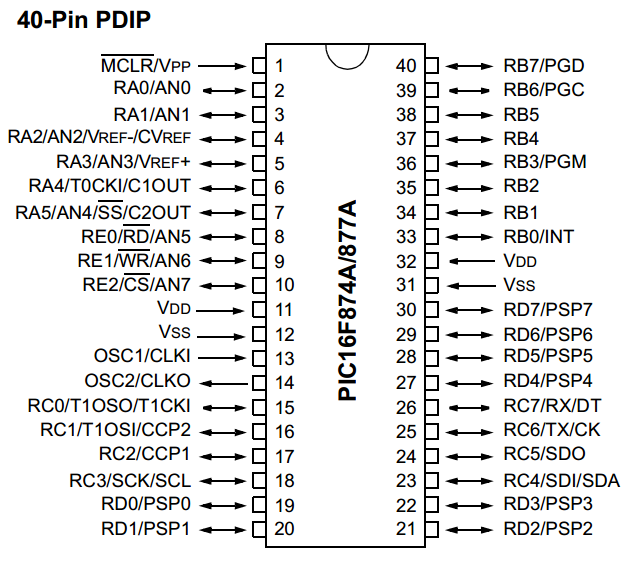
- Nguồn sử dụng: 5VDC

- Chế độ sleep

- Có 5 ports I/O (A, B, C, D, E) với 33 chân

**3.2.2 Sơ đồ chân và chức năng chân PIC16F877A**

**a) Sơ đồ chân**



*Hình 3.1: Sơ đồ chân PIC 16F877A*

**b) Chức năng các chân**

**\* Chân MCRL/Vpp(1)**

**-** MCRL: là ngõ vào reset tích cực ở mức thấp

- Vpp: Khi lập trình cho PIC thì đóng vai trò là ngõ vào nhận điện áp lập trình

**\* Chân RA0(1)/AN0(1) (2,3)**

**-** RA0(1): xuất/nhập số

- AN0(1): ngõ vào tương tự của kênh thứ 0 (1)

**\* Chân RA2/AN2/VREF-/CVREF(4)**

- RA2: xuất/nhập số

- AN2: ngõ vào tương tự của kênh thứ 2

- VREF-: ngõ vào điện áp chuẩn (thấp) của bộ A/D

- CVREF: điện áp tham chiếu VREF ngõ ra bộ so sánh

**\* Chân RA3/AN3/VREF+ (5)**

- RA3: xuất/nhập số

- AN3: ngõ vào tương tự của kênh thứ 3

- VREF+: ngõ vào điện áp chuẩn (cao) của bộ A/D

**\* Chân RA4/TOCKI/C1OUT (6)**

- RA4: xuất/nhập số

- TOCKI: ngõ vào xung clock bên ngoài cho Timer 0

- C1OUT: ngõ ra bộ so sánh 1

**\* Chân RA5/AN4/SS/C2OUT(7)**

- RA5: xuất nhập số

- AN4: ngõ vào tương tự của kênh thứ 4

- SS: ngõ vào chọn lựa SPI phụ

- C2OUT: ngõ ra bộ so sánh 2

**\* Chân RE0/RD/AN5 (8)**

- RE0: xuất nhập số

- RD: điều khiển đọc port slave song song

- AN5: ngõ vào tương tự của kênh thứ 5

**\* Chân RE1/WR/AN6 (9)**

- RE1: xuất/nhập số

- WR: điều khiển ghi port slave song song

- AN6: ngõ vào tương tự của kênh thứ 6

**\* Chân RE2/CS/AN7 (10)**

**-** RE2: xuất/ nhập số

- CS: chip chọn lựa điều khiển port slave song song

- AN7: ngõ vào tương tự của kênh thứ 7

\* **VDD (11, 32)**

**VSS (12, 31)**

**=>** là các chân nguồn của PIC

\* **Chân OSC1/CLKIN (13)**

**Chân OSC2/CLKOUT (14)**

Là các chân nối với thạch anh để tạo ra xung dao động cho PIC

**\* Chân RC0/T1OSO/T1CKI (15)**

**-** RC0: xuất/nhập số

- T1OSO: ngõ ra bộ dao động Timer 1

- T1CKI: ngõ vào xung clock bên ngoài Timer 1

\* **Chân RC1/T1OSI/CCP2 (16)**

- RC1: xuất/nhập số

- T1OSI: ngõ vào bộ dao động Timer 1

- CCP2: ngõ vào Capture 2, ngõ ra Compare 2, ngõ ra PWM 2

\* **Chân RC2/CCP1 (17)**

**-** RC2: xuất/nhập số

- CCP1: ngõ vào Capture 1, ngõ ra Compare 1, ngõ ra PWM1

\* **Chân RC3/SCK/SCL (18)**

**-** RC3: xuất/nhập số

- SCK: ngõ vào xung clock nối tiếp đồng bộ/ngõ ra của chế độ SPI

- SCL: ngõ vào xung clock nối tiếp đồng bộ/ngõ ra chế độ I2C

\* **Chân RDx/PSPx (19, 20, 21, 22, 27 – 30)**

**-** RDx: xuất/nhập số

- PSPx: dữ liệu port slave song song

\* **Chân RC6/TX/CK (25)**

- RC6: xuất/nhập số

- TX: truyền bất đồng bộ USART

- CK: xung đồng bộ USART

\* **Chân RC7/RX/DT (26)**

- RC7: xuất/nhập số

- RX: nhận bất đồng bộ USART

- DT: dữ liệu đồng bộ USART

\* **Chân RC5/SDO (24)**

**-** RC5: xuất/ nhập số

- SDO: dữ liệu ra SPI

**\* Chân RB0/INT (33)**

- RB0: xuất/nhập số

- INT: ngõ vào nhận tín hiệu ngắt ngoài

\* **Chân RB1, RB2, RB4, RB5 (34, 35, 37, 38)**

Xuất nhập số

\* **RB3/PGM (36)**

**-** RB3: xuất/nhập số

- PGM: chân cho phép lập trình điện áp thấp ICSP

\* **Chân RB6/PGC (39)**

**-** RB6: xuất/nhập số

- PGC: mạch gỡ rối và xung clock lập trình ICSP

\* **Chân RB7/PGD (40)**

**-** RB7: xuất/nhập số

- PGD: mạch gỡ rối và xung clock lập trình ICSP

**3.2.3 Các đặc tính ngoại vi**

**Có 3 bộ định thời:**

**- Timer 0:** timer/counter 8 bit, có bộ chia trước

**- Timer 1:** timer/counter 16 bit, có bộ chia trước có thể đếm khi CPU ở chế độ sleep với nguồn xung từ thạch anh hoặc nguồn xung bên ngoài

**- Timer 2**: timer/counter 8 bit, có bộ chia trước và postscale

- **2 bộ CCP (Capture/Compare/PWM)**

- Capture có độ rộng 16 bit, độ phân giải 12.5 ns

- Compare có độ rộng 16 bit, độ phân giải 200ms

- Độ phân giải lớn nhất của PWM là 10 bit

- Các chuẩn giao tiếp nối tiếp MSSP

+ Giao tiếp SPI

+ Giao tiếp I2C

- Chuẩn giao tiếp nối tiếp USART

- Chuẩn giao tiếp song song PSP

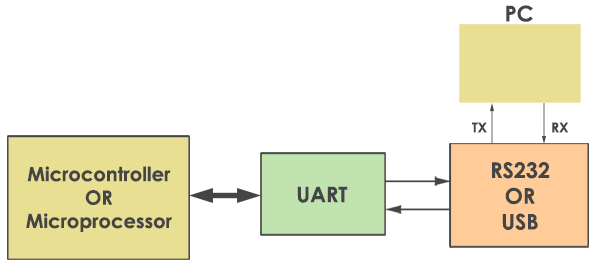
- Bộ chuyển đổi ADC với 8 kênh có độ phân giải 10 bit

- Có khả năng hoạt động với nhiều dạng Oscillator khác nhau

**3.2.4 Giao tiếp USART**

**a) Khái niệm**

**-** USART (Universal Synchronous Asynchronous Receiver Transmitter) là một trong hai module I/O nối tiếp



*Hình 3.2: Giao tiếp UART*

- Còn được gọi là giao tiếp truyền thông nối tiếp (Serial Communication Interface – SCI)

- Hai chân dùng cho USART:

+ RC6/TX/CK

+ RC7/RX/DT

- Chế độ:

+ Bất đồng bộ (Asynchronous)

+ Đồng bộ (Master mode)

+ Đồng bộ (Slave mode)

- Các dạng giao tiếp nối tiếp

+ Giao tiếp chỉ theo một chiều

+ Giao tiếp 2 chiều. Có thể giao tiếp cùng thời điểm

+ Giao tiếp 2 chiều. Không thể giao tiếp cùng thời điểm

- Tốc độ Bit( Bit rate):

+ Số lượng Bit được gửi đi trong 1giây(bps)

- Tốc độ Baud (Baud rate):

+ Số lượng kí tự được gửi trong 1 giây

+ Kí tự có thể có 1 hoặc nhiều bit

+ Với USART trong PIC 16F877A 1 kí tự = 1 bit => Bit rate ≡ Baud rate

**b) Đồng bộ**

- Truyền một khối dữ liệu (ký tự) trong cùng một thời điểm

- Giao tiếp theo chế độ half – duplex

- Một chân tạo xung clock ngoại từ master đến salve

- Chỉ sử dụng trong phạm vi ngắn

- Có thể thêm bit START, STOP, và tín hiệu chọn chip

**c) Bất đồng bộ**

**-** Truyền một byte dữ liệu trong một thời điểm

- Giao tiếp theo chế độ full – duplex

- Không có chân tạo xung clock

- Có bit START và bit STOP, bit chẳn lẻ…

- Hoạt động theo chuẩn NRZ (None – Return – to – Zero)

=> 1 bit START(0) + 8 bit DATA + 1 bit STOP (1)

- Bit LSB được truyền đi trước

- Các khối truyền và nhận dữ liệu độc lập với nhau sẽ dùng chung tần số tương ứng với tốc độ baud

- Khối truyền và nhận phải dùng chung một định dạng dữ liệu

**d) Cấu trúc thanh ghi**

**\* Thanh ghi TXSTA: Tranmit status and control register**



*Hình 3.3: Cấu trúc thanh ghi TXSTA*

**- bit 5** TXEN: Transmit Enable Bit

1: cho phép truyền

0: không cho phép truyền

**- bit 4** SYNC: USART Mode Select bit

1: dạng đồng bộ

0: dạng bất đồng bộ

**- bit 2**  BRGH: High Baud Rate Select Bit

Chỉ có tác dụng ở chế độ bất đồng bộ

1: tốc độ cao

0: tốc độ thấp

**- bit 1:** TRMT: Transmit Shift Register Status bit

1: thanh ghi TSR không có dữ liệu

0: thanh ghi TSR có dữ liệu

**- bit 7**  SPEN: Serial Port Enable Bit

1: cho phép cổng nối tiếp hoạt động

0: không cho phép cổng nối tiếp hoạt động

**\* Truyền dữ liệu:**

- Thành phần quan trọng nhất: Thanh ghi dịch TSR (Transmit Shift Register)

- Dữ liệu cần truyền sẽ được đưa trước vào thanh ghi TXREG

- TSR sẽ lấy dữ liệu từ thanh ghi đệm TXREG

+ Thanh ghi TXREG bị rỗng

+ Cờ TXIF (PIRI <4>) sẽ được set lên 1

+ Cờ TXIF không xóa được bằng phần mềm

+ Cờ TXIF chỉ được xóa khi có dữ liệu mới được đưa vào TXREG

- Trạng thái thanh ghi TSR => cờ TRMT (TXSTA <1>)

- TMRT = 1 => thanh ghi TSR rỗng

+ bit này chỉ đọc

- Thanh ghi TSR không có trong bộ nhớ dữ liệu và chỉ được điều khiển bởi CPU

- TXEN = 1: cho phép khối truyền dữ liệu hoạt động

- Quá trình truyền chỉ bắt đầu khi:

+ Có dữ liệu trong thanh ghi TXREG

+ Xung truyền baud được tạo ra

**\* Thanh ghi RCSTA: Receive status and control register**



*Hình 3.4: Cấu trúc thanh ghi RCSTA*

**- bit 7** SPEN: Serial Port Enable Bit

1: cho phép cổng nối tiếp hoạt động

0: không cho phép cổng nối tiếp hoạt động

**- bit 4**  CREN: Continous Receive Enable bit

1: cho phép nhận một chuỗi dữ liệu liên tục

0: không cho phép nhận

**- bit 2** FERR: Framing Error Bit

1: xuất hiện lỗi “framing” trong quá trình nhận dữ liệu

0: không xuất hiện lỗi

Phải kiểm tra bit FERR trước khi đọc dữ liệu mới

**- bit 1** OERR: Overrun Error bit

1: Lỗi tràn dữ liệu (Có thể xóa bằng cách xóa bit CREN)

0: không có lỗi tràn dữ liệu

\* **Nhận dữ liệu**

**-** Dữ liệu được nhận từ chân RC7/RX/DT và đi qua khối phục hồi dữ liệu

- Thực chất là một thanh ghi dịch tốc độ cao(gấp 16 hoặc 64 lần tốc độ baud)

- Thành phần quan trọng nhất: thanh ghi dịch RSR (Receive Shift Register)

- Bit điều khiển cho phép nhận dữ liệu: CREN (RCSTA <4>)

- Dữ liệu nhận được trong thanh ghi RSR sẽ được đưa vào thanh ghi RCREG => cờ RCIF = 1 (RCIF = 0 khi dữ liệu tại RCREG đã được đọc)

- RCREG là thanh ghi có bộ đếm kép hoạt động theo cơ chế FIFO (First In First Out)

+ RCREG có thể nhận 2 byte dữ liệu

+ Byte thứ 3 tiếp tục đưa vào thanh ghi RSR

- Nếu bit STOP của byte thứ 3 đã được nhận mà thanh ghi RCREG vẫn đầy:

+ Cờ báo tràn OERR sẽ được set

+ Dữ liệu trong thanh ghi RSR sẽ bị mất

+ Quá trình đưa dữ liệu từ RSR đến RCREG sẽ bị gián đoạn

=> Phải lấy hết dữ liệu của thanh ghi RCREG trước khi nhận dữ liệu tiếp theo

- Nếu OERR = 1 => quá trình chuyển dữ liệu từ RSR đến RCREG sẽ bị cấm

- Bit OERR phải được xóa bằng phần mềm: xóa bit CREN rồi set lại

- Bit FERR và bit dữ liệu thứ 9 (RX9D) được đưa vào bộ đệm như 8 bit dữ liệu

- Đọc thanh ghi RCREG => Cập nhật giá trị mới cho FERR và RX9D

=> Đọc thanh ghi RCSTA trước khi đọc thanh ghi RCREG

**3.2.5 Giới thiệu về giao tiếp UART – USB TTL**

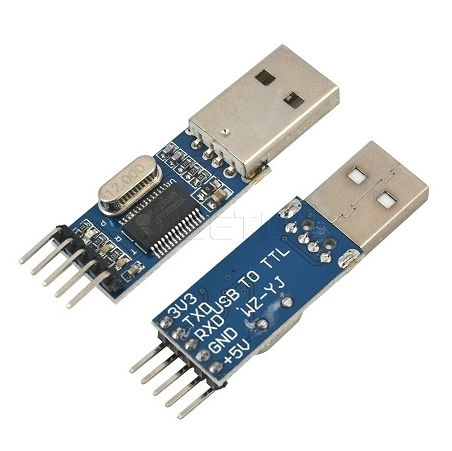
**a) Chuẩn UART (hay chuẩn TTL)**

**-** Chuyển 1 bit trong một đơn vị thời gian theo một tốc độ truyền nhận dữ liệu quy định (tốc độ baud)

- Mức “0” => 0V

- Mức “1” => từ 3.3V tới 5V

**b) Giới thiệu về USB to TTL Serial PL 2303HX**



*Hình 3.5: USB TTL PL2303HX*

**- USB to TTL Serial PL2303HX sử dụng IC chuyển đổi USB to COM PL2303HX với nhiều cải tiến về hiệu năng của hãng Prolific.**

- Module USB to TTL Serial **PL2303HX** có sẵn các thành phần cần thiết, chỉ cần cắm vào cổng USB của máy tính là có ngay 1 cổng COM ở mức 5V có thể giao tiếp với vi điều khiển.

- Module USB to TTL Serial có 3 LED gồm:

+ 1 **LED**nguồn

+ 1 **LED** báo tín hiệu Tx

+ 1 **LED** báo tín hiệu Rx.

- Module PL2303HX đã có sẵn đầu ra 3.3V giành cho các IC điện áp thấp. Sản phẩm được bán kèm theo 4 dây 2 đầu cái (female-female) để tiện trong kết nối với mạch ngoài.

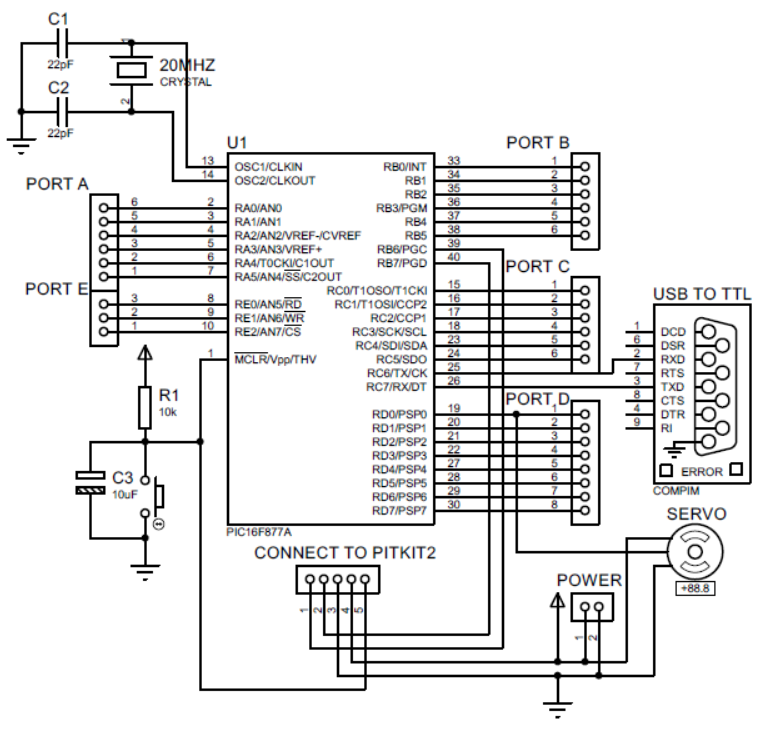
**\* Sơ đồ các chân trên module USB to TTL Serial PL2303HX:**

|  |  |
| --- | --- |
| Chân Module PL2302HX | Chân Vi điều khiển |
| GND | GND |
| TX | RX |
| RX | TX |
| 5V | Vcc |
| 3.3V |  |

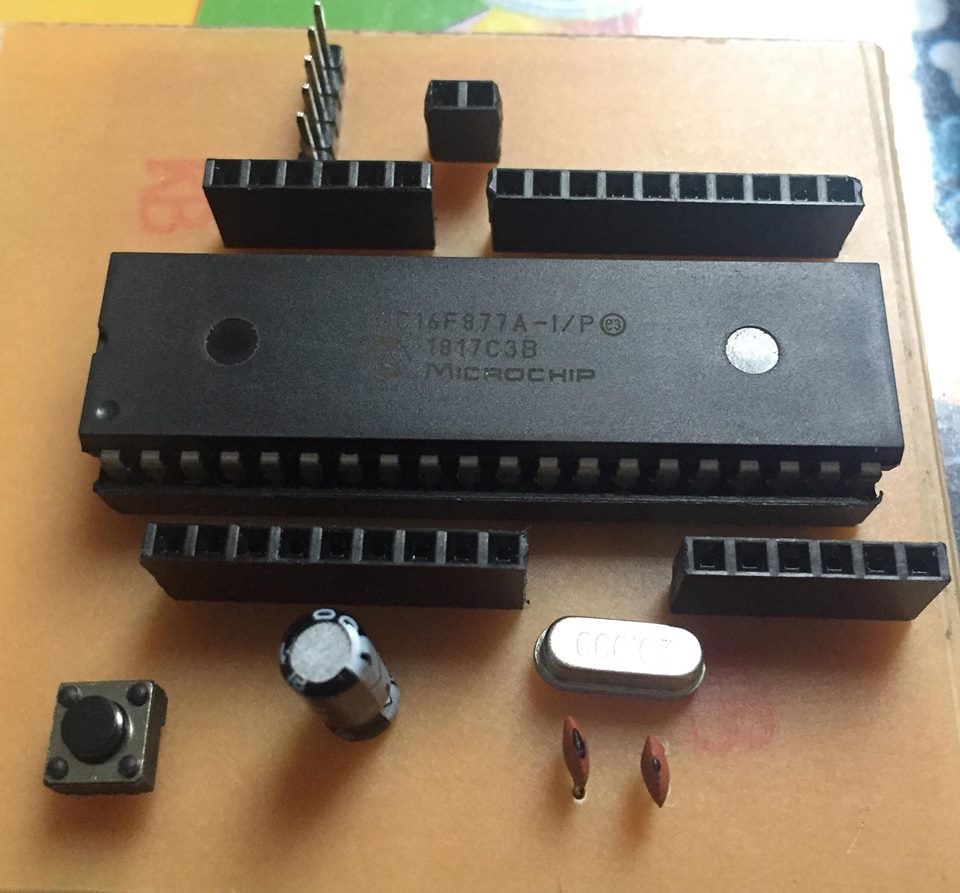
**CHƯƠNG 4: ĐIỀU KHIỂN HỆ THỐNG BĂNG TẢI PHÂN LOẠI HÌNH DẠNG SẢN PHẨM BẰNG XỮ LÝ ẢNH**

**4.1 Sơ đồ nguyên lý và nguyên lý hoạt động của mạch điều khiển**

**4.1.1 Sơ đồ mạch nguyên lý**



*Hình 4.1: Sơ đồ nguyên lý mạch điều khiển*



*Hình 4.2: Sơ đồ mạch thật PIC*

**4.1.2 Nguyên lý hoạt động**

- Ở đây PIC giao tiếp với phần mềm Matlab của máy tính thông qua cổng COM. Khi nhận tín hiệu nó sẽ thực hiện lệnh đã được lập trình sẵn có trong PIC.

- Khi có tín hiệu ‘0’ (tức nhận biết hình tròn) gửi về thì động cơ Servo quay một góc 135o

- Khi có tín hiệu ‘1’ (tức nhận biết hình vuông) gửi về thì động cơ Servo quay một góc 90o

- Khi có tín hiệu ‘2’ (tức nhận biết hình tam giác) gửi về thì động cơ Servo quay một góc 45o

**4.1.3 Lưu đồ thuật toán**

Xử lý ảnh

C = 0

< 0.27

ĐC Servo

Quay 135o

Đ

S

C = 1

< 0.27

ĐC Servo

Quay 90o

Đ

S

ĐC Servo

Quay 45o

Đ

C = 2

< 0.27

S

ĐC Servo

Không quay

**4.1.4 Code PIC**

#include <doanpic.h>

#use rs232(baud=9600,parity=N, xmit=PIN\_C6, rcv=PIN\_C7, bits=8)

CHAR c;

void main()

{

WHILE (TRUE)

{

c = getch () ;

IF (c == '0')

{

for (INT i = 0; i < 10; i++)

{

output\_high (PIN\_d0) ;

delay\_us (1900) ;

output\_low (PIN\_d0) ;

delay\_us (18100) ;

}

}

IF (c == '1')

{

for (INT i = 0; i < 10; i++)

{

output\_high (PIN\_d0) ;

delay\_us (1500) ;

output\_low (PIN\_d0) ;

delay\_us (18500) ;

}

}

IF (c == '2')

{

for (INT i = 0; i < 10; i++)

{

output\_high (PIN\_d0) ;

delay\_us (1000) ;

output\_low (PIN\_d0) ;

delay\_us (19000) ;

}

}

}

}

**4.2 Quá trình xử lý ảnh và code Matlab**

**4.2.1 Ý tưởng:**

**-** Đã có nhiều đề tài xử lý ảnh với các ý tưởng khác nhau. Qua quá trình tìm hiểu các tài liệu, các phương thức xử lý ảnh, nhóm đã hình thành ý tưởng chính cho đề tài nhận diện hình dạng sản phẩm như sau:

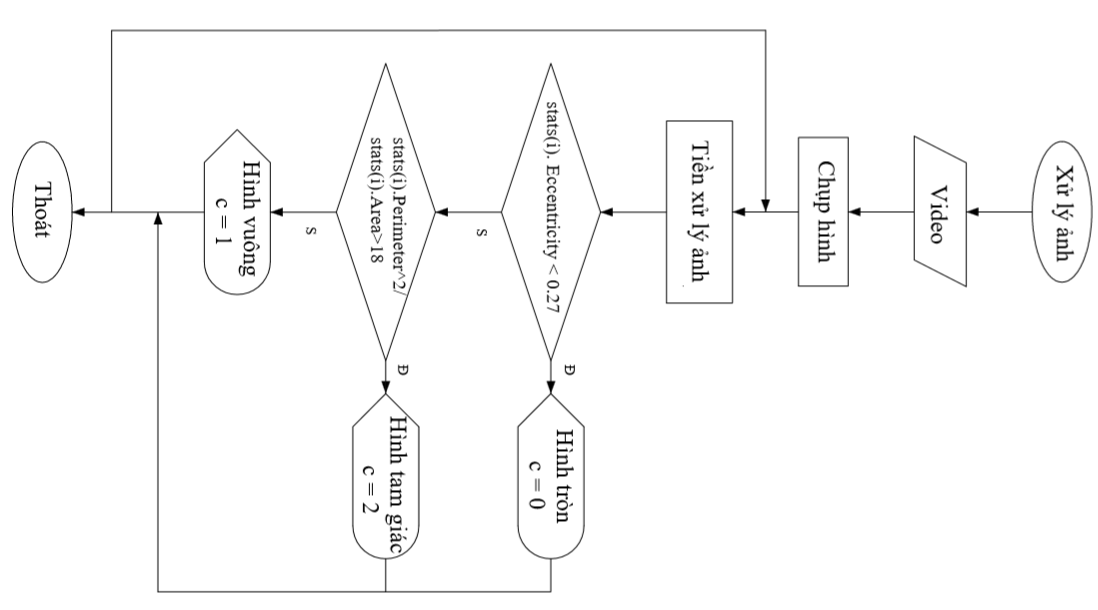
+ Bước 1: Chụp ảnh từ camera: sử dụng camera có độ phân giải

+ Bước 2: Tiến hành xử lý ảnh chụp được từ camera: chuyển đổi ảnh, xóa nhiều, tăng độ tương phản, lấy biên, xác định tọa độ tâm…

+ Bước 3: Dùng các hàm xác định độ tròn, so sánh diện tích để nhận dạng được hình dáng của sản phẩm.

- Để đạt được kết quả thì quá trình khử nhiễu đóng vai trò rất quan trọng nên các thông số được lựa chọn tỉ mỉ.

**4.2.2 Lưu đồ thuật toán xử lý ảnh**



**4.2.3 Code chương trình Matlab**

function varargout = doanvdk(varargin)

% DOANVDK MATLAB code for doanvdk.fig

% DOANVDK, by itself, creates a new DOANVDK or raises the existing

% singleton\*.

%

% H = DOANVDK returns the handle to a new DOANVDK or the handle to

% the existing singleton\*.

%

% DOANVDK('CALLBACK',hObject,eventData,handles,...) calls the local

% function named CALLBACK in DOANVDK.M with the given input arguments.

%

% DOANVDK('Property','Value',...) creates a new DOANVDK or raises the

% existing singleton\*. Starting from the left, property value pairs are

% applied to the GUI before doanvdk\_OpeningFcn gets called. An

% unrecognized property name or invalid value makes property application

% stop. All inputs are passed to doanvdk\_OpeningFcn via varargin.

%

% \*See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one

% instance to run (singleton)".

%

% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help doanvdk

% Last Modified by GUIDE v2.5 22-Nov-2018 12:53:26

% Begin initialization code - DO NOT EDIT

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @doanvdk\_OpeningFcn, ...

'gui\_OutputFcn', @doanvdk\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

% End initialization code - DO NOT EDIT

% --- Executes just before doanvdk is made visible.

function doanvdk\_OpeningFcn(hObject, eventdata, handles, varargin)

% This function has no output args, see OutputFcn.

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% varargin command line arguments to doanvdk (see VARARGIN)

% Choose default command line output for doanvdk

handles.output = hObject;

% Update handles structure

guidata(hObject, handles);

% UIWAIT makes doanvdk wait for user response (see UIRESUME)

% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.

function varargout = doanvdk\_OutputFcn(hObject, eventdata, handles)

% varargout cell array for returning output args (see VARARGOUT);

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure

varargout{1} = handles.output;

% --- Executes on button press in start.

function start\_Callback(hObject, eventdata, handles)

% hObject handle to start (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

global img obj stop s

stop = 0;

obj = videoinput('winvideo',1);

obj.ReturnedColorSpace = 'rgb';

hImage = image(zeros(150,170,3),'Parent',handles.axes1);

preview(obj, hImage);

while 1

if stop == 1

break;

end

img = getsnapshot(obj);

pause(0.1);

axes(handles.axes2);

imshow(img);

b = rgb2gray(img);

c = im2bw(b,graythresh(b));

c = bwareaopen(c,3000);

sel = strel('disk',2);

c = imclose(c, sel);

c = imfill(c, 'holes');

[label,num] = bwlabel(c,8);

stats = regionprops(label,'all');

if(num < 10)

for i = 1:num

if stats(i).Eccentricity <0.27

area =stats(i).Area;

bankinh = sqrt(area/pi);

tam = stats(i).Centroid;

t = 0:0.1:2\*pi;

x = tam(1) + bankinh\*cos(t);

y = tam(2) + bankinh\*sin(t);

hold on

plot(x,y,'r');

plot(tam(1),tam(2),'r+');

text(tam(1)-10,tam(2)-20,'hinh tron ','BackgroundColor',[.7 .9 .7]);

fwrite(s,'0');

set(handles.text1,'string','HINH TRON');

pause(0.5);

hold off

else

thisboundingbox = stats(i).BoundingBox;

if stats(i). Area>10000

rectangle('Position', [thisboundingbox(1), thisboundingbox(2), thisboundingbox(3), thisboundingbox(4)], 'EdgeColor','r','LineWidth',2);

else

rectangle('Position', [thisboundingbox(1), thisboundingbox(2), thisboundingbox(3), thisboundingbox(4)], 'EdgeColor','b','LineWidth',2);

end

if stats(i). Perimeter^2/stats(i).Area>18

text(stats(i). Centroid(1),stats(i).Centroid(2), 'tamgiac','Color','r');

fwrite(s,'2');

set(handles.text1,'string','TAM GIAC');

else

text(stats(i).Centroid(1),stats(i).Centroid(2), 'vuong', 'Color', 'g');

fwrite(s,'1');

set(handles.text1,'string','VUONG');

end

end

end

end

end

% --- Executes on button press in stop.

function stop\_Callback(hObject, eventdata, handles)

% hObject handle to stop (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

global stop

stop=1;

close;

% --- Executes on button press in connect.

function connect\_Callback(hObject, eventdata, handles)

% hObject handle to connect (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

global s

s=serial('COM5');

fopen(s);

% --- Executes on button press in disconnect.

function disconnect\_Callback(hObject, eventdata, handles)

% hObject handle to disconnect (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

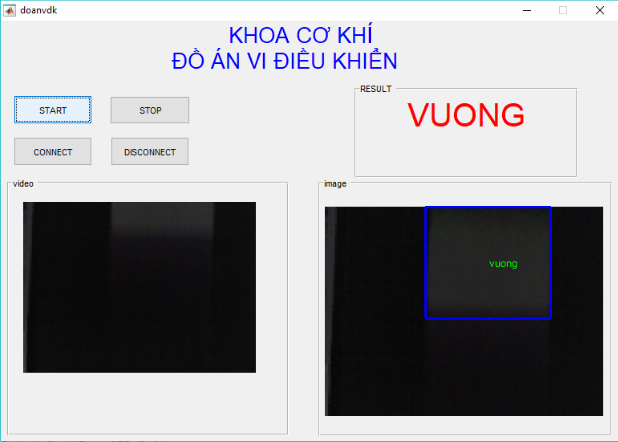
global s

fclose(s);

**4.3 Mô phỏng thực tế**

****

*Hình 4.3: Nhận dạng hình tam giác*

****

*Hình 4.4: Nhận dạng hình vuông*



*Hình 4.5: Nhận dạng hình tròn*