DESARROLLO DE UN JUEGO PARA NIÑOS UTILIZANDO LA BIBLIOTECA OPEN CV EN DETECCION COLORES.

Franz Paccha, Rubén Báez

Universidad Técnica Particular de Loja

f-a-p-c-1999@hotmail.com--rubenbaaez1995@gmail.com

RESUMEN- El juego consiste ayudar a niños a desarrollar sus destrezas .Por eso hemos decidido crear un juego matemático mediante el uso de un computador y una cámara, para que los niños con un rango de edad entre 5 y 7 años encuentren como alternativa, un método de aprendizaje que les permita interactuar con un computador haciendo más dinámico y entretenido el aprendizaje así mismo facilitando su comprensión en operaciones matemáticas, y también agilizando su mente para dar respuestas rápidas y acertadas.

I. INTRODUCCIÓN

El seguimiento de objetos es una de las tareas más relevantes dentro de visión por computador, así mismo de la biblioteca openCV. Actualmente existe mucha competencia en este campo pesto que se pueden desarrollar programas muy útiles, así como son la detección de rostros, lo cual lo encontramos incluso en nuestros smartphones.

II. MARCO TEORICO

Para la detección de colores hemos decidido utilizar el modelo de color HSV (Matiz, Saturación, Valor)

Los pasos que hemos seguido para poder detectar los colores mediante HSV son los siguientes:

1. Obtenemos la imagen de nuestra webcam.
2. Convertimos esa imagen a el formato de color HSV (Matiz, Saturación, Brillo)
3. Especificamos un rango de color para crear una plantilla, de blanco o negro, 0 y 1, verdadero falso... etc. (para ello nosotros utilizaremos barras de configuración)
4. Localizamos en la ventana el lugar que ocupa ese color en ese frame
5. Dibujamos un circulo en esa posición.

De la libreria OpenCV usaremos estas nuevas funciones:

- cv2.createTrackbar ('Titulo', 'ventana', rango inferior, rango superior, variable temporal).

Crea una barra deslizante.

- cv2.cvtColor (imagen, Tipo de conversión) Convierte la imagen de una paleta de colores a otra

- cv2.inRange(imagen, Rango min, Rango max)

- cv2.moments.

Datos sobre la posición de un objeto donde:

-m00 -> Área de blanco

-m01 -> Posición x

-m10 -> Posición y

- cv2.erode -> erosiona la imagen para eliminar puntos erróneos

- cv2.dilate -> dilata los pixeles detectados

- cv2.circle (imagen, posición, color, tamaño de la línea)

Mostrar la imagen

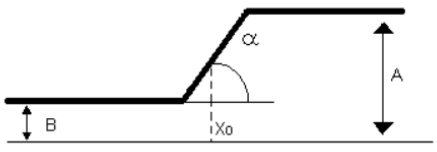
Mostraremos dos ventanas. En la primera aparecerá la imagen original con el centro del objeto. La segunda será la máscara en blanco y negro.

1. cv2.imshow('mask', mask)
2. cv2.imshow('Camara', imagen)

Una de las técnicas más utilizadas es la de detección de bordes

Para obtener la forma de la imagen a partir de la cual se calcula el BSM es necesario obtener los bordes o contorno de la imagen. Los bordes de una imagen [6] se pueden definir como transiciones entre dos regiones de niveles de

gris significativamente distintos. Estos nos facilitan una valiosa información sobre las fronteras de los objetos que nos interesan reconocer del resto de la imagen.



La figura 1 muestra un modelo unidimensional y continuo de un borde. Este modelo representa una rampa desde un nivel de gris bajo B a uno alto A.H corresponde a la variación de la intensidad. Se calcula de la siguiente forma

H = A-B

donde α corresponde al ángulo de inclinación de la rampa α y X0 corresponde a la coordenada horizontal X0 donde se encuentra el punto medio de la rampa. Un operador que proporcionara los valores de X0 y H daría unos datos muy valiosos sobre la imagen, ya que proporcionaría la amplitud del borde, y la localizaría con exactitud dentro de la imagen.

III. DESARROLLO DE LA APLICACIÓN

1. Requerimientos

Para el desarrollo de esta aplicación se establecen algunos requerimientos como:

- Tener instalada la biblioteca openCV phyton
- Una cámara web
- Un entorno donde se pueda ejecutar el programa

2. Desarrollo

Esta aplicación está basada en detección de colores, convirtiéndolos en números los cuales mediante operaciones matemáticas sacaremos un resultado el cual deberá coincidir con el numero ya antes mostrado en pantalla.

El proceso inicia con el filtrado del color de interés del objeto a detectar en la imagen (ver figura 1).

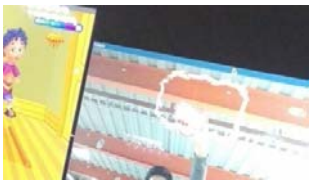


Figura1: Detección del objeto

A continuación, se convierte la imagen en el espacio de color HSV (ver figura 2), seguidamente se usa el filtro cvInRange dos veces para filtrar los colores de interés en este caso azul (ver figura 3).

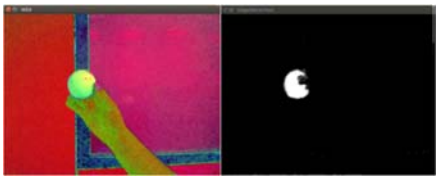


Figura2: transformación de RGB a HSV

Seguidamente utilizamos un recuadro en la pantalla que se va a mostrar en el funcionamiento del juego para que cuando ingrese un color dentro de el nos muestre un numero el cual luego se va a sumar con algún otro color.

Resultados:

El resultado es bastante bueno puesto que permite tener una distancia de al menos 2 metros para que el programa funcione normalmente, pero se recomienda tener objetos con colores vivos para que facilite la detección del color, también tener un lugar donde no entre el sol y tenga buena iluminación

COLOR	Iluminación		Distancia (cámara-esfera)	Movimiento (esfera)	
	Alta	*Baja*	*1m*	*Lento*	*Rápido*
Amarillo	✓	✓	✓	✓	✓
Verde	✓	✓	✓	✓	✓
Azul	✓	✓	✓	✓	✓

IV. CONCLUSIONES

 La aplicación está dirigida a niños, con el propósito de mejorar su aprendisaje

 La aplicación se desarrolló para detectar los colores de objetos en tiempo real para los colores azul, rojo y amarillo obteniendo muy buenos resultados.

 La aplicación es muy fiable en un rango de 1 a 2m puesto que la luz distorsiona un poco el color, y mientras mas lejos estés más difícil será para la web cam identificar estos colores.

REFERENCIAS

http://geekyhour.blogspot.com/2014/11/opencv-y-python-ii-seguimiento-de.html

http://acodigo.blogspot.com/2016/04/seguimiento-de-objetos-por-color.html

https://robologs.net/2014/07/02/deteccion-de-colores-con-opencv-y-python/