# **PIXY**

Em 1999, a CREATE Lab, uma divisão do Instituto de Robótica da Universidade Carnegie Mellon, EUA), lançou a CMUcam, uma câmera com um sensor que facilitava a criação de aplicações que utilizassem de visão computacional. A CMUcam se tornou amplamente conhecida pelos entusiastas da área e facilitou o trabalho daqueles que trabalhavam na área, permitindo que projetos utilizando visão se tornassem menos complexos e trabalhosos. Com o passar dos anos novas versões da CMUcam foram sendo lançadas trazendo atualizações e melhorias na câmera, aperfeiçoando seu funcionamento, e facilitando seu uso. Em 2014, na quinta versão da CMUcam (também conhecida como CMUcam5), uma campanha bem-sucedida no Kickstarter e uma parceria com Charmed Labs gerou uma nova formulação da câmera lançada resultando na Pixy.

Em uma parceria com a Charmed Labs e uma boa campanha no Kickstarter, a CMUcam5 se tornou Pixy, mantendo a qualidade da CMUcam mas trazendo um nome mais fácil e curto. Sendo a sucessora da CMUcam, a Pixy também tem como objetivo facilitar o desenvolvimento de aplicações que utilizem visão e sensores em seus projetos. A Pixy é plug-and-play e possui compatibilidade com Arduino, Raspberry PI, aplicações LEGO, dentre outros microcontroladores. A sucessora da Pixy original, a Pixy2, foi lançada sendo um modelo menor e mais sofisticado da câmera.

**CONECTANDO A PIXYCAM EM UMA PLACA ARDUINO**

A PixyCam possui suporte para conexão com diversos dispositivos, desde microcontroladores até dispositivos LEGO. Isso torna a PixyCam muito versátil e amplifica seu uso, já que pode ser aplicada em diferentes plataformas. E por ser um dispositivo plug-and-play (onde não são necessários ajustes para seu funcionamento imediato, sendo só conectar e usar) a sua utilização se torna dinâmica e simplificada.

A PixyCam, ao ser conectada ao computador interage com um programa chamado PixyMon. Este programa é responsável pelas configurações da câmera, e exibe na tela as imagens que estão sendo registradas pela câmera, e o reconhecimento em si. Através desse programa podemos fazer a configuração da câmera, definindo a luminosidade, quantidade de frames por segundo, quantidade de pixels a serem analisados, qualidade da imagem, dentre diversos outros ajustes para definir um melhor reconhecimento dos objetos. Além dessas configurações, o PixyMon também faz a definição dos objetos a serem reconhecidos, definindo-os com assinaturas utilizadas para identificar os objetos.

A interação entre a câmera e a placa é bem simples. Construímos o código desejado dentro da IDE do Arduino carregando os plugins necessários para dentro do software (todos os plugins, firmware, e softwares necessários são disponibilizados no site da PixyCam) e carregamos os códigos para dentro da placa. A programação da placa para interagir com a câmera é bem simples e são necessários poucos comandos para conseguir as informações dos objetos reconhecidos. A ideia da interação entre a placa e a câmera é coletar as informações dos objetos reconhecidos pela PixyCam. Tais informações são controladas pelo PixyMon, que é a ponte entre a câmera e a placa.

Resumidamente, conectamos a câmera no Arduino e no computador. A câmera interage com um programa que controla o reconhecimento dos objetos, as configurações da câmera, e as assinaturas dos objetos reconhecidos. Os códigos na placa acessam as informações de posicionamento, assinatura, e demais dados dos objetos reconhecidos, e trabalha em cima dessas informações dependendo dos métodos implementados no código. A seguir temos um exemplo de código onde coletamos os dados dos objetos reconhecidos e os exibimos no terminal:

Código 1 Exibindo os objetos reconhecidos

#include <Pixy2.h>

// Criamos o objeto da PixyCam

Pixy2 pixy**;**

void setup**(){**

// Definimos a saída no terminal

Serial**.**begin**(**115200**);**

// Iniciamos a câmera

pixy**.**init**();**

**}**

void loop**(){**

int i**;**

// Pega os objetos detectados

pixy**.**ccc**.**getBlocks**();**

// Se existirem objetos reconhecidos eles são exibidos no terminal

**if** **(**pixy**.**ccc**.**numBlocks**){**

Serial**.**print**(**"Detectado: "**);**

Serial**.**println**(**pixy**.**ccc**.**numBlocks**);**

**for** **(**i **=** 0**;** i **<** pixy**.**ccc**.**numBlocks**;** i**++){**

Serial**.**print**(**"\n Bloco "**);**

Serial**.**print**(**i**);**

Serial**.**print**(**": "**);**

// Exibe todas as informações do objeto detectado

pixy**.**ccc**.**blocks**[**i**].**print**();**

**}**

**}**

**}**

A programação necessária para a utilização das funcionalidades de reconhecimento da câmera é bem específica, apesar de ser extremamente simples. A seguir temos alguns dos principais comandos para a utilização da câmera:

* pixy.ccc.blocks[i].m\_signature Retorna a assinatura (o tipo do objeto definido nas configurações de reconhecimento da câmera) do objeto rastreado;
* pixy.ccc.blocks[i].m\_x Retorna a posição x do objeto rastreado;
* pixy.ccc.blocks[i].m\_y Retorna a posição y do objeto rastreado;
* pixy.ccc.blocks[i].m\_width Retorna a largura do objeto rastreado;
* pixy.ccc.blocks[i].m\_height Retorna a altura do objeto rastreado;
* pixy.ccc.blocks[i].m\_angle Retorna o ângulo do objeto rastreado;
* pixy.ccc.blocks[i].m\_index Retorna o index do objeto rastreado;
* pixy.ccc.blocks[i].m\_age Retorna o número de frames que o objeto está sendo rastreado;
* pixy.ccc.blocks[i].print() Exibe as informações do objeto.

**CONFIGURANDO A PIXYCAM (ISSO TU VÊ DEPOIS PQ HOJE NÃO VAI ROLAR MAS NEM FODENDO)**

**SCRATCH (ESCREVE SÓ UM POUCO PQ NÉ, NÃO FAZ NEM SENTIDO TU ESCREVER MUITA COISA SOBRE ESSA BOSTA)**

Scratch 😊

# **SCRATCH VIDEO SENSING (DETALHAR EM QUAIS APLICAÇÕES O SCRATCH VIDEO SENSING FUNCIONA BEM VER ONDE COLOCAR)**

O Scratch fornece uma biblioteca para a utilização da webcam em seus projetos junto com os demais blocos, esse plugin é chamado de Video Sensing. A Video Sensing permite que o usuário interaja com o programa através da sua webcam e possa realizar eventos no jogo dependendo de qual objeto ele esteja interagindo. Apesar de muito interessante e abrir uma porta para a utilização dessa tecnologia em diferentes aplicações, esse plugin ainda é bastante limitado.

**APLICAÇÃO DO JOGO EM SCRATCH E PIXYCAM (SÓ UMA BÁSICA PQ NÃO TEM PQ TU FICAR SE MATANDO DE ESCREVER ISSO AGORA)**

A ideia central do Coopera é utilizar de visão computacional para o reconhecimento de objetos e aplicar essa tecnologia no desenvolvimento de um jogo sério ativo para o estimulo da cooperação e colaboração entre crianças. Tendo isso em mente a utilização da PixyCam se torna muito pertinente pois, em teoria, facilitaria o desenvolvimento da aplicação.

# **CONECTANDO O SCRATCH COM O ARDUINO**

Para a interação entre o computador e a PixyCam (em um contexto mais amplo, podendo utilizar dos dados disponibilizados pela câmera) é necessário um dispositivo auxiliar conectado tanto no computador, quanto na placa. Como visto anteriormente, um dos dispositivos auxiliares pode ser uma placa Arduino, com a qual a Pixy tem uma interação fácil de se realizar.

Tendo em vista a possibilidade de usar placas Arduino para executarem o papel de suporte da câmera, e tendo que são dispositivos bastante acessíveis, e o Scratch possui distribuições que interagem com placas Arduino, surgiu-se a ideia de tentar utilizar essas placas para a interação do Scratch com a PixyCam.

## ScratchX (EXPLICAR DIREITO COMO FUNCIONA E TAL)

Existe uma versão beta chamada ScratchX que traz plugins que ainda estão em fase de teste. Dentro do ScratchX existem diversas bibliotecas para serem utilizadas juntamente com os blocos padrões do Scratch. As possibilidades dentro do ScratchX são gigantes, pois pode-se incluir diversos conjuntos de blocos ainda em testes. Em contrapartida, a versão do Scratch utilizada por essa modificação é mais antiga. O ScratchX é totalmente online e para sua utilização é só acessar o site.

## Scratch for Arduino (S4A) (EXPLICAR DIREITO COMO FUNCIONA E TAL)

Scratch for Arduino (S4A) é uma distribuição do Scratch voltada para a utilização do Arduino. A distribuição conta com diversos blocos customizados para a interação dos blocos de código padrões do Scratch com as funções da placa Arduino. Os blocos oferecem funcionalidades desde as mais básicas até as mais complexas. O S4A possui compatibilidade com as versões Diecimila, Duemilanove e Uno. Outras versões não foram testadas, mas devem funcionar normalmente. O S4A possui compatibilidade com aplicações Scratch padrões, logo, pode abrir projetos feitos no mesmo. Porém, não é possível compartilhar projetos em S4A no sistema de jogos do Scratch.

# **CLARIFAI (EXPLICAR DIREITO COMO FUNCIONA E TAL)**

Em contrapartida a utilização da PixyCam surgiu a ideia de utilizar o Clarifai. A Clarifai é uma empresa que fornece APIs para soluções em análises e reconhecimento de imagens. A tecnologia utiliza de um sistema de treinamento para reconhecimento de objetos onde você cria uma aplicação e define os objetos a serem reconhecidos através de exemplos. A cada objeto adicionado a identificação dos mesmos fica mais treinada e aumentado cada vez mais sua precisão. Como dito anteriormente, para a utilização do Clarifai você cria uma aplicação, define os objetos a serem escaneados (não há um limite para a quantia de objetos que cada aplicação pode armazenar) e faz o treinamento da aplicação através de imagens semelhantes e filtragem de acertos e erros. A criação das aplicações é muito simples e não há um limite para a quantia de aplicações que podem ser criadas.

# REALIZANDO A CONEXÃO ENTRE O SCRATCH E A PIXYCAM

Apesar do Scratch possuir alguns métodos de realizar a conexão com o Arduino, a aplicação da PixyCam utilizando a placa junto com a ferramenta de desenvolvimento se torna aparentemente inviável. A interação realizada entre a placa e o Scratch ocorre quase que totalmente no controle dos pinos lógicos e analógicos. Sendo que a proposta da conexão entre o Scratch e o Arduino seria executar o controle dessas portas, controlar o fluxo de energia, dentre outras funcionalidades, onde a proposta central é programar a placa, sem usar linhas de código.

A programação necessária para a utilização das funcionalidades de reconhecimento da câmera é bem específica, apesar de ser extremamente simples. Os códigos são feitos dentro da IDE do Arduino e carregados para placa. Temos os seguintes comandos para a utilização da PixyCam no Arduino:

* pixy.ccc.blocks[i].m\_signature Retorna a assinatura (o tipo do objeto definido nas configurações de reconhecimento da câmera) do objeto rastreado;
* pixy.ccc.blocks[i].m\_x Retorna a posição x do objeto rastreado;
* pixy.ccc.blocks[i].m\_y Retorna a posição y do objeto rastreado;
* pixy.ccc.blocks[i].m\_width Retorna a largura do objeto rastreado;
* pixy.ccc.blocks[i].m\_height Retorna a altura do objeto rastreado;
* pixy.ccc.blocks[i].m\_angle Retorna o ângulo do objeto rastreado;
* pixy.ccc.blocks[i].m\_index Retorna o index do objeto rastreado;
* pixy.ccc.blocks[i].m\_age Retorna o número de frames que o objeto está sendo rastreado;
* pixy.ccc.blocks[i].print() Exibe as informações do objeto.

Como pudemos ver, a utilizações da biblioteca Pixy para interação com a PixyCam2 é bem simples e intuitiva. Porém, a utilização dessa biblioteca em qualquer um dos métodos de conexão entre o Arduino e o Scratch é impossível, pois, não há como inserir linhas de código, muito menos importar a biblioteca para podermos utilizar os comandos. Portanto, apesar da conexão funcionar e ser possível utilizar dela para diversos fins, ela interage mais diretamente com as portas e não permite uma abrangência maior no uso da placa, o que inviabiliza o uso de tais métodos.

O principal problema em utilizar dos métodos já existentes para a conexão do Arduino com o Scratch é que não podemos passar variáveis do Arduino para os blocos do Scratch. Para entendermos melhor é preciso explicar o que a placa realmente faz ao interagir com a câmera e todo o funcionamento em si.

A PixyCam, ao ser conectada ao computador (para o funcionamento da câmera, esta precisa estar conectada tanto na placa quanto no computador) interage com um programa chamado PixyMon. Este programa é responsável pelas configurações da câmera, e exibe na tela as imagens que estão sendo registradas pela câmera, e o reconhecimento em si. Através desse programa podemos fazer a configuração da câmera, definindo a luminosidade, quantidade de frames por segundo, quantidade de pixels a serem analisados, qualidade da imagem, dentre diversos outros ajustes para definir um melhor reconhecimento dos objetos. Além dessas configurações, o PixyMon também faz a definição dos objetos a serem reconhecidos, definindo-os com assinaturas utilizadas para identificar os objetos.

A interação entre a câmera e a placa é bem simples. Construímos o código desejado dentro da IDE do Arduino carregando os plugins necessários para dentro do software (todos os plugins, firmware, e softwares necessários são disponibilizados no site da PixyCam) e carregamos os códigos para dentro da placa. Como dito anteriormente, a programação da placa para interagir com a câmera é bem simples e são necessários poucos comandos para conseguir as informações dos objetos reconhecidos. A ideia da interação entre a placa e a câmera é coletar as informações dos objetos reconhecidos pela PixyCam. Tais informações são controladas pelo PixyMon, que é a ponte entre a câmera e a placa.

Resumidamente então temos que, conectamos a câmera no Arduino e no computador. A câmera interage com um programa que controla o reconhecimento dos objetos, as configurações da câmera, e as assinaturas dos objetos reconhecidos. Os códigos na placa acessam as informações de posicionamento, assinatura, e demais dados dos objetos reconhecidos, e trabalha em cima dessas informações dependendo dos métodos implementados no código. A seguir temos um exemplo de código onde coletamos os dados dos objetos reconhecidos e os exibimos no terminal:

Como pudemos ver, a conexão entre a placa e a câmera se dá de uma maneira bem específica através dos códigos, e esta interação não é possível pelos métodos convencionais de interação disponíveis anteriormente citados, que tratam da interação em um âmbito mais geral.

Código 2 Exibindo os objetos reconhecidos

#include <Pixy2.h>

// Criamos o objeto da PixyCam

Pixy2 pixy**;**

void setup**(){**

// Definimos a saída no terminal

Serial**.**begin**(**115200**);**

// Iniciamos a câmera

pixy**.**init**();**

**}**

void loop**(){**

int i**;**

// Pega os objetos detectados

pixy**.**ccc**.**getBlocks**();**

// Se existirem objetos reconhecidos eles são exibidos no terminal

**if** **(**pixy**.**ccc**.**numBlocks**){**

Serial**.**print**(**"Detectado: "**);**

Serial**.**println**(**pixy**.**ccc**.**numBlocks**);**

**for** **(**i **=** 0**;** i **<** pixy**.**ccc**.**numBlocks**;** i**++){**

Serial**.**print**(**"\n Bloco "**);**

Serial**.**print**(**i**);**

Serial**.**print**(**": "**);**

// Exibe todas as informações do objeto detectado

pixy**.**ccc**.**blocks**[**i**].**print**();**

**}**

**}**

**}**