

SE 3XA3: MIS OpenCameraRefined

Team #211, CAMERACREW
Faisal Jaffer, jaffem1
Dominik Buszowiecki, buszowid
Pedram Yazdinia, yazdinip
Zayed Sheet, sheetz

April 4, 2020

Table 1: **Revision History**

Date	Version	Notes
March 14, 2020	1.0	Initial document
April 4, 2020	2.0	Initial document

Gesture Controller Module

Module

GestureController

Uses

TensorflowObjectDetectionAPI

ClassifierConstants

CameraController

Filter

Classifier

Syntax

Exported Constants

N/A

Exported Types

GestureController = ?

Exported Access Programs

Routine name	In	Out	Exceptions
GestureController		GestureController	classifier_initialize_failure
processImage			
setFrame	<i>byte</i> []		
captureImage			
showFilter			

Semantics

State Variables

imageFrame: *byte*[] # current camera frame

classifier: *Classifier* # TensforFlow model

filter: *Filter*

recognition: *Recognition*[] # classification on the current frame

Environment Variables

None

State Invariant

None

Assumptions

- The mathematical operator \backslash represents integer division. For example $8 \backslash 5 = 1$.

Access Routine Semantics

GestureController():

- transition: $classifier, filter := Classifier(ClassifierConstants.getInferenceName(), ClassifierConstants.getLabelName()), Filter()$
- output: $out := self$
- exception: $exc := ((ClassifierConstants.getInferenceName() == null \vee ClassifierConstants.getLabelName() == null) \Rightarrow classifier_initialize_failure)$

processImage():

- transition: $recognition := classifier.recognizeImage(bitmapFromByte(imageFrame)) \Rightarrow (recognition.title == ClassifierConstants.Smile \rightarrow captureImage() \mid recognition.title == ClassifierConstants.Thumb \rightarrow filter.changeFilter())$
- output: None
- exception: None

setFrame(*byte*[] data):

- transition: $imageFrame := data$

- output: None
- exception: None

captureImage():

- transition: *CameraController.captureImage()*
- output: None
- exception: None

showFilter():

- transition: None
- output: None
- exception: None

Local Functions

bitmapFromByte(*byte*[] data):

- transition: Matrix mat := ImageUtils.convertYUV420SPToARGB8888(data)
- output: *out* := mat
- exception: None

Recognition Module

Module

Recognition

Uses

RectF

Syntax

Exported Constants

N/A

Exported Types

Recognition = ?

Exported Access Programs

Routine name	In	Out	Exceptions
Recognition	<i>String, String, \mathbb{Q}, RectF</i>	Recognition	
getID		<i>String</i>	
getTitle		<i>String</i>	
getConfidence		\mathbb{Q}	
getLocation	<i>Bitmap</i>	<i>RectF</i>	

Semantics

State Variables

id: *String* # unique id assignment since multiple recognitions possible

title: *String* # label of the recognition

confidence: \mathbb{Q} # confidence level of the classification

location: RectF # pixel location of the recognition

Environment Variables

None

State Invariant

None

Assumptions

- Invalid arguments will not be provided into the Recognition and getLocation routines.

Access Routine Semantics

Recognition(*id*, *title*, *confidence*, *location*):

- transition: $id, title, confidence, location := id, title, confidence, location$
- output: $out := self$
- exception: None

getID():

- transition: None
- output: $out := id$
- exception: None

getTitle():

- transition: None
- output: $out := title$
- exception: None

getConfidence():

- transition: None
- output: $out := confidence$
- exception: None

getLocation():

- transition: None
- output: $location$
- exception: None

Local Functions

None

Classifier Module

Module

Classifier

Uses

Recognition

ClassifierConstants

Bitmap

TF

Syntax

Exported Constants

N/A

Exported Types

None

Exported Access Programs

Routine name	In	Out	Exceptions
Classifier	<i>String, String, (\mathbb{Z}, \mathbb{Z})</i>	Classifier	classifier_initialize_failure
recognizeImage	<i>Bitmap</i>	<i>Recognition</i>	

Semantics

State Variables

modelFilename: *String* # link to the model inference

labelFilename: *String* # link to the model labels

inputSize: (\mathbb{Z} , \mathbb{Z}) # size of model input

model: TF.model# stored model

Environment Variables

None

State Invariant

None

Assumptions

- Invalid arguments will not be provided to the Classifier and recognizeImage routines.

Access Routine Semantics

Classifier(modelFilename, labelFilename, inputSize):

- transition: $model := newTF.model(modelFilename, labelFilename)$
- output: $out := self$
- exception: $exc := newTF.model(modelFilename, labelFilename) == null \Rightarrow (classifier_initialize_failure)$

recognizeImage(b):

- transition: None
- output: $out := model.detect(b)$
- exception: None

Local Functions

None

Classifier Constants Module

Module

ClassifierConstants

Uses

Syntax

Exported Constants

inferenceName: *String*

labelName: *String*

Exported Types

None

Exported Access Programs

Routine name	In	Out	Exceptions
getInferenceName		<i>String</i>	
getLabelName		<i>String</i>	

Semantics

State Variables

Environment Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

getInferenceName():

- transition: None
- output: $out := inferenceName$
- exception: None

getLabelName():

- transition: None
- output: $out := labelName$
- exception: None

Local Functions

None

Filter Module

Module

Filter

Uses

Filter Constants

Syntax

Exported Constants

N/A

Exported Types

N/A

Exported Access Programs

Routine name	In	Out	Exceptions
changeFilter			
getFilter		\mathbb{Z}	
setFrame	<i>byte[]</i>		
processImage			
getFiltered		<i>Bitmap</i>	

Semantics

State Variables

filterIndex : \mathbb{Z} # Which index in the *FILTERS* constant from the *filters* module is selected

imageFrame: *byte[]* # current camera frame

rgbFrameBitmap: *Bitmap* # bitMap version of camera frame

filtered: *Bitmap* # Filtered version of rgbFrameBitmap

Environment Variables

preview: # Android Camera Preview object

State Invariant

$\text{filterIndex} < |\text{FILTERS}|$

Assumptions

The % operator represents the mathematical modulus operator

Access Routine Semantics

`changeFilter()`:

- transition: $\text{filterIndex} := (\text{filterIndex} + 1) \% |\text{FILTERS}| + 1$
- output: None

`getFilter()`:

- transition: None
- output: $\text{out} := \text{filterIndex}$

`setFrame(byte[] data)`:

- transition: $\text{imageFrame} := \text{data}$
- output: None
- exception: None

`processImage()`:

- transition:
 $\text{filterIndex} = 0 \implies \text{filtered} := \text{null}$
 $\text{filterIndex} = 0 \implies \text{filtered} := \text{setFiltered}(\text{rgbFrameBitmap})$
- output: None
- exception: None

`getFiltered()`:

- transition: None
- output: $\text{out} := \text{filtered}$

Filter Constants Module

Module

Constants

Uses

OpenCV

Syntax

Exported Constants

GRAYSCALE: $(\mathbb{Z}[], \mathbb{Z}[])$ ColorMatrixColorFilter

RED_FILTER: $(\mathbb{Z}[], \mathbb{Z}[])$ ColorMatrixColorFilter

BLUE_FILTER: $(\mathbb{Z}[], \mathbb{Z}[])$ ColorMatrixColorFilter

~~*# The values of the filters will be obtained from the OpenCV lookup tables. They are 2xN matrices which*~~

~~*# The values of the filters are obtained from code snippets found online, the type ColorMatrixColorFilter is a type built into android*~~

FILTERS = [GRAYSCALE, RED_FILTER, BLUE_FILTER]

Exported Types

None

Exported Access Programs

None

Semantics

State Variables

None

State Invariant

None