Project

Look how you spell it!

Look how you spell it! Is a game in augmented reality that interacts with the printed game of the same name that teaches the child above 5 years to read and write words of their daily life besides also teaching other languages.

When pointing the smartphone, tablet or camera of a computer on the calligraphy blade that accompanies the game, the animal in question appears on the blade.

Children are motivated to write correctly so that animals suggested by games can "create life." Once written the word, COW for example, a kitty in 3D, caricatured, appears and emits its characteristic sound - MUUUUU.

The goal of the game is to motivate the child to want to learn to read and write and to make them feel more pleasure in learning. Initially we will launch the theme with animals, then we will go to other themes, such as space, dinosaurs and even playful themes with the characters they like the most.

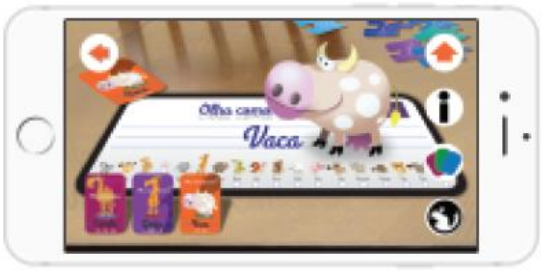

Figure: After the child writes the animal's name on the card, an animated 3D projection appears with the sound of the little animal and the card related to it appears as conquered.

Project's goal: So far, we have the game in printed form, the word recognition module of the digital game is the purpose of this work.

It will involve:
Word Recognition Step

1) Collect letters in the School using a form

   i) Extract the letters from the form and clean them

2) Train the OCR using SVM

3) Card Detection and Word Recognition

   i) Card localization using landmarks

   ii) Change of perspective - to correct the perspective of the card

   iii) Segmentation - to segment the letters of the word on the board

   iv) Word recognition using the SVM trained

Steps of the Development

Introduction

The project initially provided for the use of a public database to do OCR, but I did not find any that was useful for this project. So I decided to make my own base of letters and numbers.

I went to my daughter's school and got 107 children to collect the letters and written cards. They are children from 4 to 7 years old.

For convenience, I also decided to make all scripts in python and then move to C ++ and IOS / Android

If the prototype is considered satisfactory for the game, I will have to collect letters from more children from other schools to better train the OCR.

System Configuration:
- SO -- Windows 10
- OpenCV 3.3
- Python 3.6

1- Collect letters in the School using a form

They were collected letters from 107 children in the Anglo Literacy school. The children are from 4 to 7 years old.

I've created the form showed in the figure 1 to collect the letters:

Then I did a script for automatically cut all letters in the 107 forms, "detect_Rect.py" in the **detectRecLetters-Form** folder.

I got 2760 letters from A-Z. Some letters were discarded due to errors in children's writing. The numbers were collected but not used.

The figure 2 shows a general fluxograma used to detect the rectangles and get the letters.

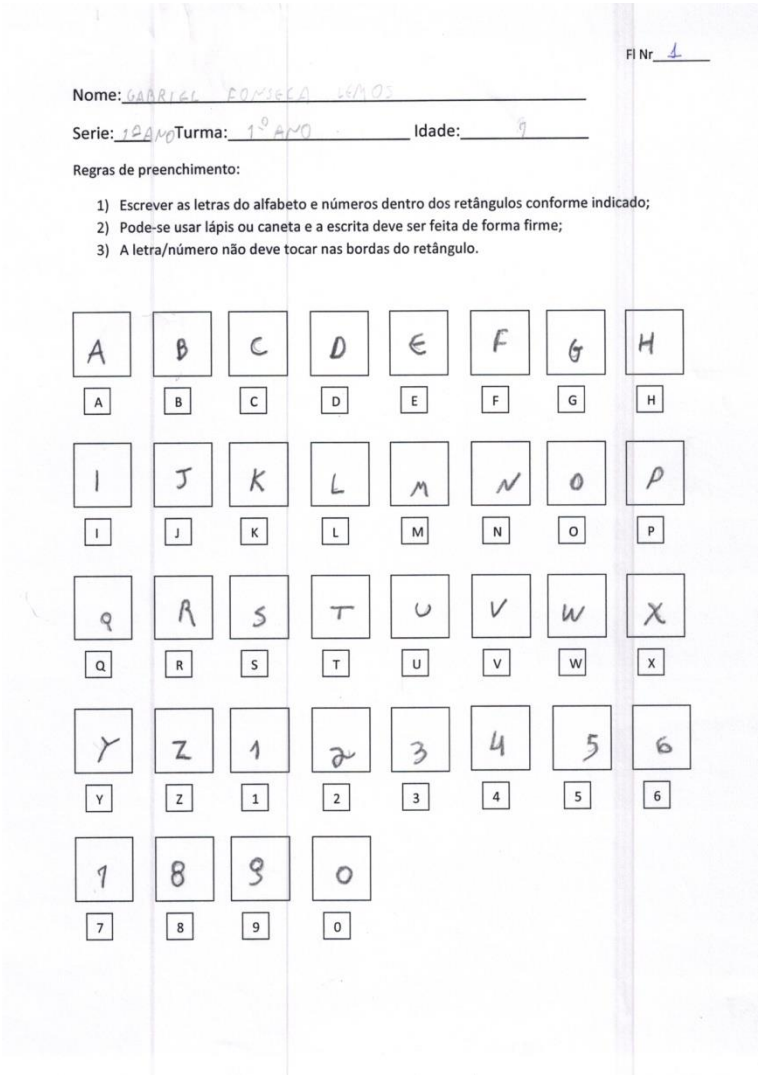

FIGURE 1: Example of the form used to collect letters in the School.

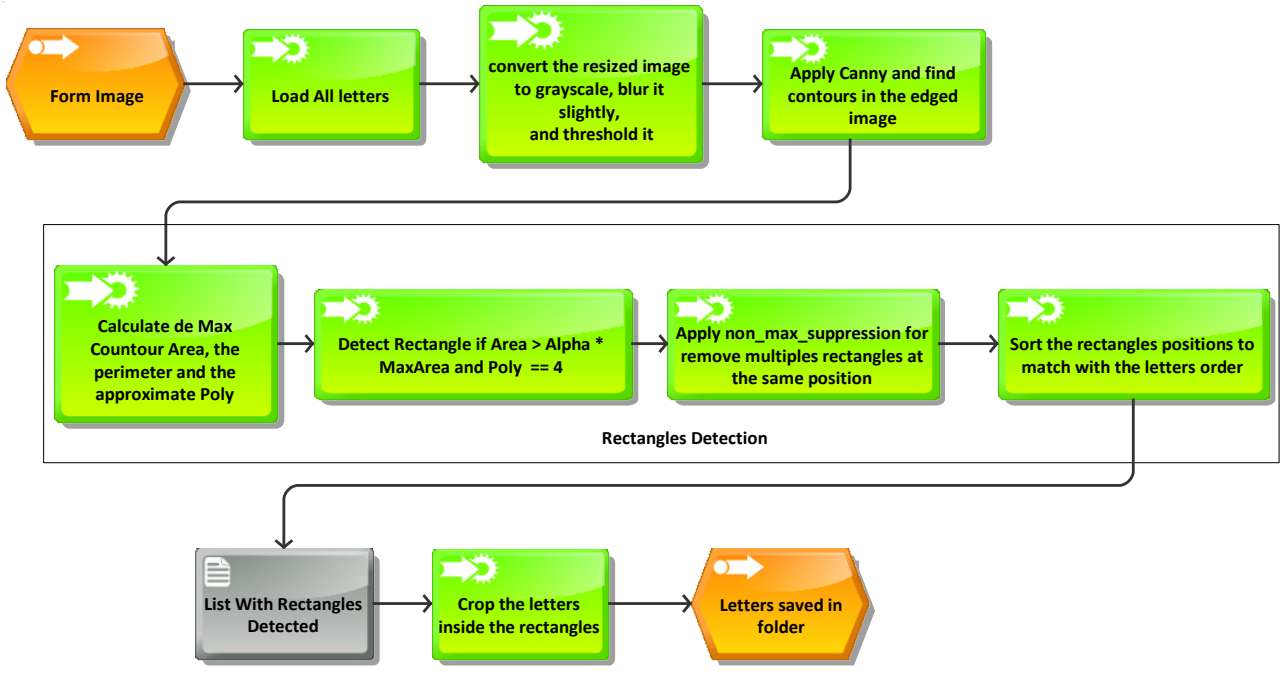

FIGURE 2: General Fluxogram used to detect letters from the form

2- Train the OCR using SVM

I did the OCR training with HOG and SVM. I used the “train_letters.py” script in the **trainSVM-letters** folder.

Database:
Total: 2760 letters
Test: 286
Train: 2474

Evaluation – Accuracy rate: 91.96 %

The figure 3 shows the general fuxogram for this task. Table 1 shows the confusion matrix for this training.

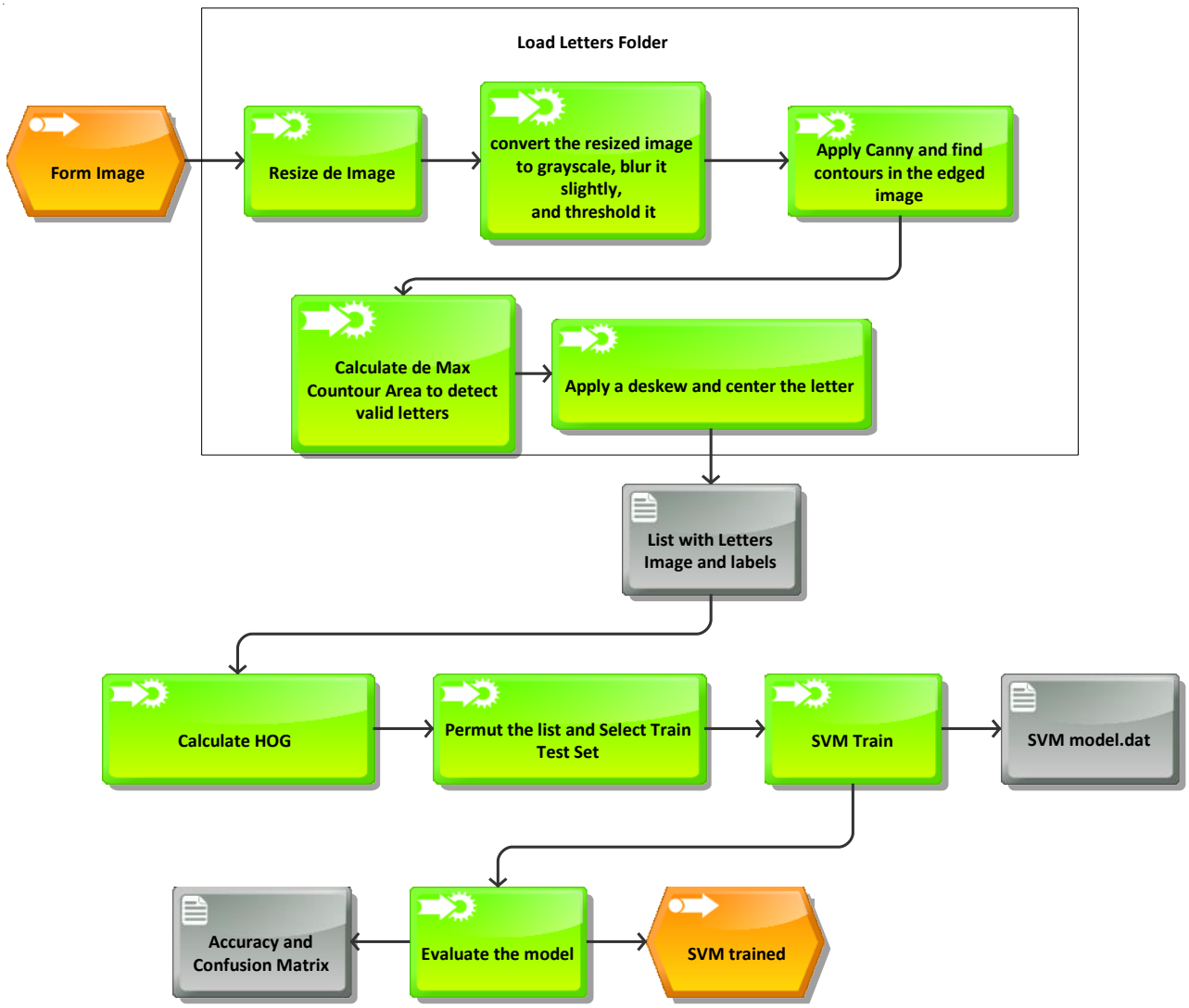

FIGURE 3: General Fluxogram used to training de SVM

TABLE 1: Confusion Matrix for letter recognition using SVM

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
"A"	10	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
"B"	0	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
"C"	0	0	9	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
"D"	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
"E"	0	0	0	0	10	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
"F"	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
"G"	0	0	0	0	1	0	9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
"H"	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
"I"	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
"J"	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
"K"	0	0	0	0	0	0	0	0	0	0	9	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0
"L"	0	0	0	0	0	0	0	0	1	0	0	8	0	0	0	0	0	0	1	1	0	0	0	0	0	0
"M"	0	0	0	0	0	0	0	1	0	0	0	1	9	0	0	0	0	0	0	0	0	0	0	0	0	0
"N"	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	1	0	0	0
"O"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0
"P"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	10	0	0	0	0	0	0	0	0	0	0
"Q"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	10	0	0	0	0	0	0	0	0	0
"R"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0
"S"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	1	0	0	0	0	0
"T"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0
"U"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	1	0	0	0	0
"V"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	10	0	0	0	0
"W"	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	10	0	0	0
"X"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0
"Y"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	10	0
"Z"	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11

3- Step 3 – Card Detection and Word Recognition

At this stage, I try to identify the words in the calligraphy cards. I asked the children to write the names of the animals as indicated on the pictures on the card.

The goal is to locate and cut the letters for use in OCR. In order to locate the card in the video, I used BRISK as the feature and FLANN to match. After detecting the card, I made a perspective correction to cut the writing area of the card and locate the letters on it.

I used the “detectLetters_video.py” script in the **detectLetters-Card** folder. Figure 4 shows an exemple of Cards that were collected and figure 5 shows the fluxograma used.

FIGURE 4: Example of Cards that were collected

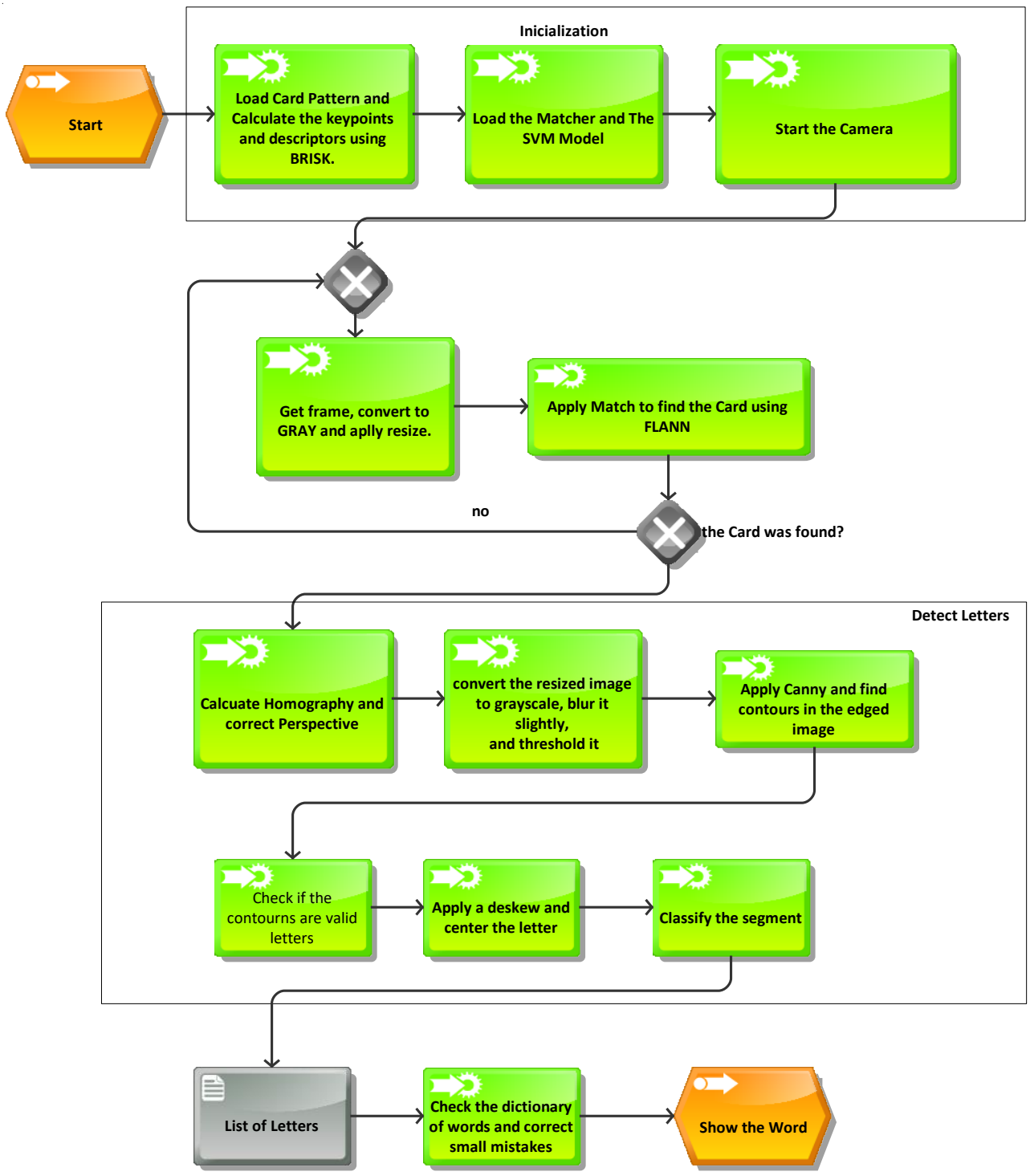

FIGURE 5: General Fluxogram for the word recognition.

Next Steps:

a) Improve the error control
b) Improve the letters segmentation on the Card. Sometimes, when the letters are close together, the system doesn't recognise correctely the word.
c) Improve the câmera stabilization
d) Move all code to C++ for iOS and Android.