In-vehicle baby alert system

Advanced Digital Image Processing project

F. Casciola, E. G. Ceroni, N. Landolfi

Università degli Studi di Siena

date TBD

Introduction

Vehicular heatstroke is largely underestimated by the general public. The majority of parents are misinformed and likely to believe that they could **never forget** their child in a vehicle.

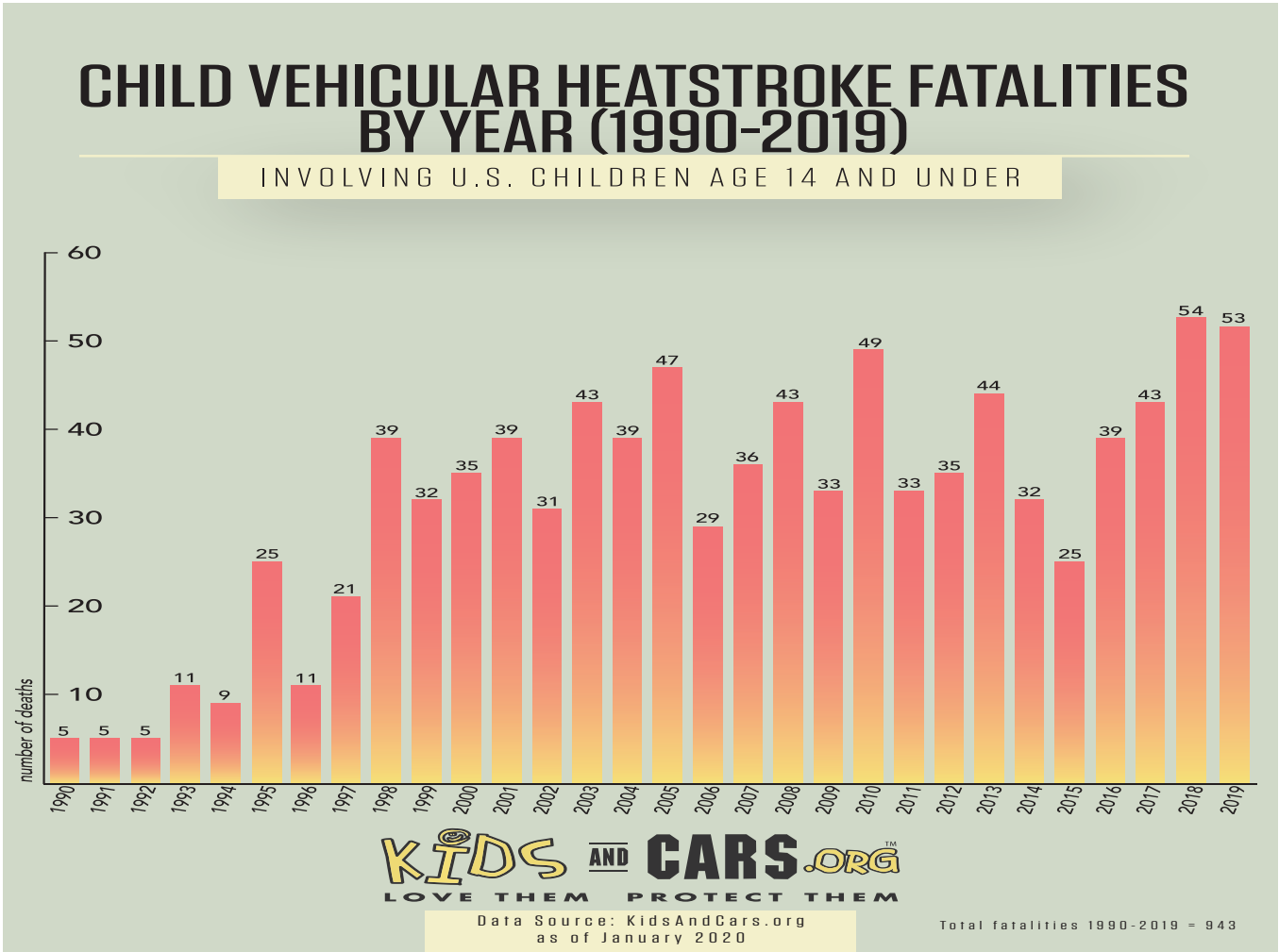
In over 55% of these cases, the person responsible for the child's death unknowingly left them in the vehicle. The most dangerous mistake one can make is to think leaving a child alone in a vehicle could never happen to them.

Introduction

The inside of a vehicle heats up very quickly! Even with the windows cracked, the temperature inside a car can reach **51 degrees Celsius** in minutes.

A child's body overheats three to five times faster compared to an adult, and heatstroke occurs when the body's temperature exceeds 40 degrees Celsius and the body organs begin to shut down.

Introduction: Some Data



Introduction: Project Proposal

Based on what we have learned about Computer Vision and Image Processing, a possible solution would be to design a new system which enables **adult/child's face detection**.
Even better, a hybrid solution which combines several way of measuring/sensing the child's presence would be more robust.

Our proposal is composed of three main steps:

- Collecting the data and building a dataset;
- Model selection and synthetic testing;
- Field testing of the best model.

The Dataset: Collecting The Data

This is the most challenging part of the project. Getting pictures of children under the age of 3 years old is not that easy.

In the beginning, we scraped images from Google Images, but we opted for a pre-existing licensed dataset[1].

[1] **eidinger2014age**.

The Dataset: Sub-sampling And Dataset Adjustments

Since the pre-existing dataset is designed for a multi-class age classification task, we applied sub-sampling.

This yields an equal number of samples for adults and children, thus focusing the problem on a **binary classification task**.

Moreover, we decided that the images should mostly contain faces with as little background as possible. To this end, we fed our images into a face extractor[2].

[2] We settled for MTCNN over HAAR cascade.

Dataset - Definitive Version

Eventually, the dataset has been split in:

- Training set: 3520 child faces and 3624 adult faces
- Validation set: 379 child faces and 401 adult faces
- Test set: 387 child faces and 238 adult faces

Use-Case Overview

The classification task consists of 3 steps:

1. Image acquisition from a USB camera (e.g Logitech C270);
2. Face extraction with MTCNN;
3. Classification of the extracted faces.

Face extractor

As mentioned above, we used a face extractor for two reasons:

- Training set creation: labeling faces by hand was too slow and tedious
- Extraction of faces from the acquired image (main use case)

We began with HAAR cascade, both frontal and lateral, then switched to MTCNN, which proved far superior.

MTCNN

Framework:

- Image resizing for the creation of a piramid of images
- The image piramid is fed to three different CNNs:
  1. First, the **P-Net** produces a large number of candidate BBs[3] and performs BB regression, followed by NMS[4] for merging the overlapping ones.
  2. Surviving candidates are fed to the **R-net** that performs BB regression and again NMS.
  3. At last, the survived boxes are fed to the **Q-net** that performs similarly to the R-net but it is more complicated and outputs the positions of **five facial landmarks**.

[3] BB = Bounding box

[4] Non-maximum suppression

MTCNN

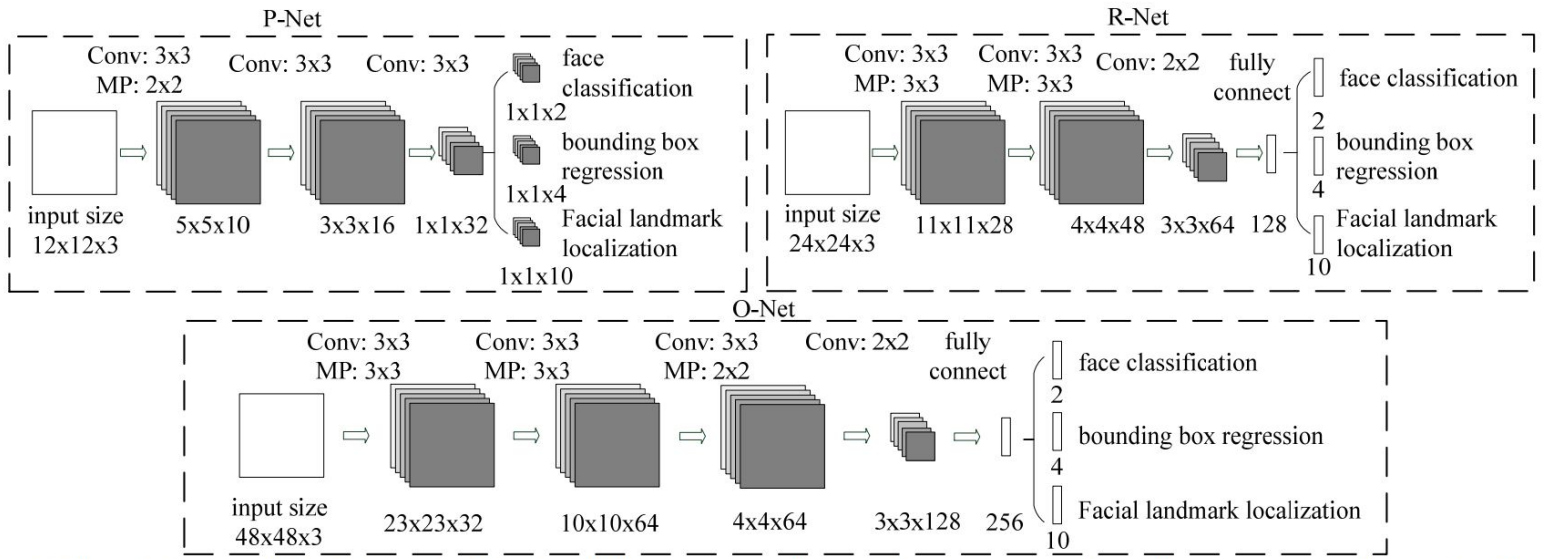


Fig. 2. The architectures of P-Net, R-Net, and O-Net, where "MP" means max pooling and "Conv" means convolution. The step size in convolution and pooling is 1 and 2, respectively.

Figure: MTCNN schematics[5].

[5] **zhang2016joint**.

Fisherface: Generalities

Given the real-time nature of the problem we are facing, we decided to try and use also a leaner method to determine whether a face in the image belongs to a child or an adult: Fisherface.

Each sample is flattened into a single vector during the training phase. Then, a PCA[6] is performed over all the dataset, in order to find a new basis to represent our data with a reduced dimensionality[7].

[6] Principal Component Analysis.
[7] **Belhumeur1997**.

Fisherface: Generalities

Once the dataset is projected onto the new basis, we use it to compute the covariance matrix and the mean vector for each class.

The covariance matrices and the mean vectors are used to find a generalized eigenvector onto which we can project the data achieving a good separation between the projections related to different classes.

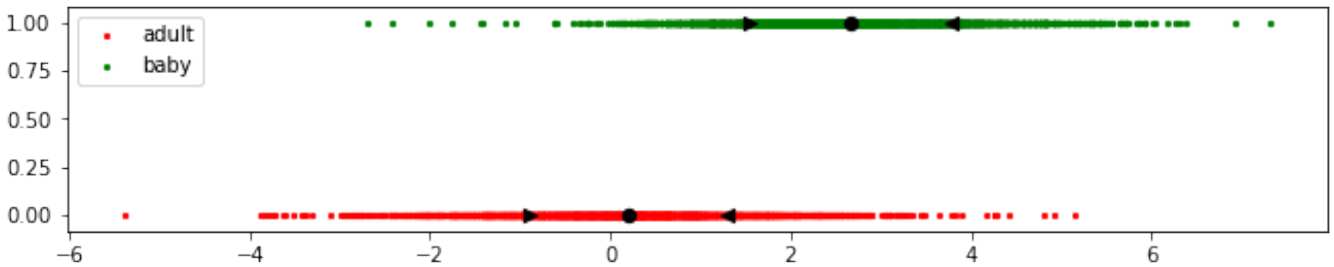


Figure: Values of the projection onto the generalized eigenvector. The triangles indicate the first standard deviation w.r.t the mean.

Fisherface: Generalities

Computing the dot product between the basis resulting from the PCA and the generalized eigenvector, we obtain a transformation which projects the flattened pictures directly on the latter. This makes the method particularly suitable for our problem.

In our use case, the pictures extracted by the MTCNN will only have to go through the following steps:

- The picture must be resized and flattened, then;
- Projected onto the generalized eigenvector, and finally;
- The projection has to be compared with a threshold T (if it is $> T$ it will be classified as child, otherwise as adult).

Fisherface: The Threshold

According to the Fisher's linear discriminant theory, the threshold T that best separates the data is

$$T = \frac{\boldsymbol{w} \cdot (\boldsymbol{\mu}_0 + \boldsymbol{\mu}_1)}{2},$$

where $\boldsymbol{w}$ is the generalized eigenvector and $\boldsymbol{\mu}_0, \boldsymbol{\mu}_1$ are the means of the two classes after they've been projected on the principal components' basis.

The following table shows the results (on the test set) for discriminating adults and children, using different picture (input) sizes and $n \in \{100, 200, 300, 400, 500\}$ principal components.

Fisherface: Test table

Input Size	Principal Components	Accuracy on Children	Accuracy on Adults	Threshold
100x100	500	88,63	72,27	1,485
	400	88,37	71,43	1,477
	300	89,14	70,17	1,448
	200	89,4	70,59	1,415
	100	88,37	68,48	1,394
50x50	500	88,11	72,27	1,484
	400	88,89	72,27	1,467
	300	88,89	70,17	1,450
	200	88,63	70,17	1,408
	100	88,63	68,07	1,392
30x30	500	87,08	70,17	1,491
	400	86,56	72,27	1,472
	300	**88,63**	**73,11**	1,435
	200	88,37	72,26	1,409
	100	89,15	67,65	1,388

Siamese Neural Network: Introduction

As previously mentioned, age classification is a challenging problem due to the complexity of the features that make up a face.

So we chose a **discriminative** approach, since we want to be able to separate **children** from **non-children**.

This was achieved by taking advantage of a **Siamese neural network**[8] that takes two inputs: a **template** image and the input image from the face extractor and checks if they belong to the same class or not.

[8] Actually there is only one network that is used to process the two inputs.

Siamese Neural Network - The Pairs

This kind of neural networks require in input a pair of images:

- Template image: the class example
- Input image: the image that has to be classified

The label $(1, 0)$ symbolizes that the template image and the input image belong to the same class, $(0, 1)$ otherwise.

Siamese Neural Network - The Pairs

We selected 26 child images as templates and paired them with all the other images in the original dataset[9], obtaining a new larger set of samples. The same procedure has been done with the adults images.

```
Creating Datasets

Training set:
Number of same class image pairs = 95391, Number of different class image pairs = 97848, total sample pairs: 193239

Validation set:
Number of same class image pairs = 10584, Number of different class image pairs = 10827, total sample pairs: 21411

Test set:
Number of same class image pairs = 10800, Number of different class image pairs = 6426, total sample pairs: 17226
```

Figure: Siamese training - validation - test set (children network)

[9] We excluded the pairs which contained the same image

Siamese Neural Network - General Architecture

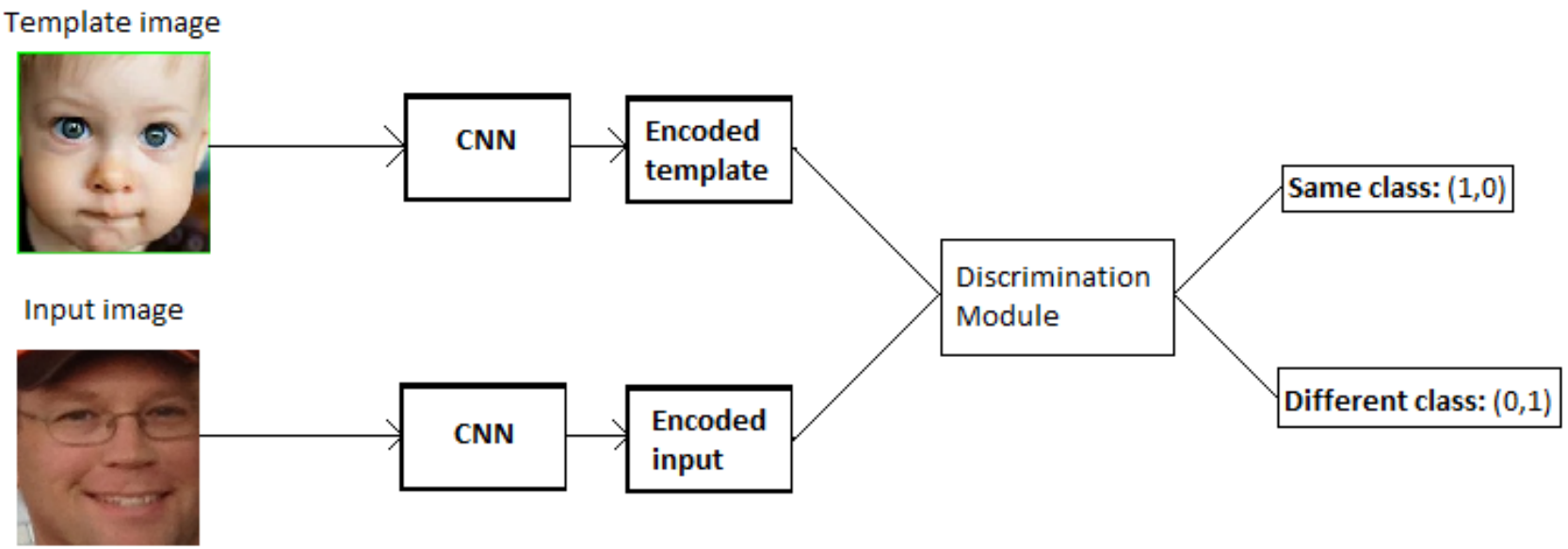


Figure: General outline of the network, the CNN is the same for both images

Siamese Neural Network - Discrimination module

The CNN part of the system actually works as an image encoder, extracting features from both the template and the input image [10], which are then fed to the **discrimination module**.
The paper[11] that inspired this approach used a joining neuron that calculated the cosine distance between the encoded vectors.
We decided to implement two different discriminator modules, one based on the **euclidean distance** between the CNN-encoded vectors and for the other one a **multi-layered perceptron** which was fed the concatenation of the two encoded vectors.

[10] Could be optimized at runtime by preprocessing the templates
[11] **bromley1994signature**.

Siamese Neural Network - Model selection strategies

We began our work by implementing a modified, slimmed-down version of the VGG16 architecture, based on the remarkable results that this model obtained in **ILSVRC**[12] 2014.

However, we were not satisfied with the results, so we decided to build a custom network and started the cross-validation process. This however was taking too long even for a small subset of hyperparameters (although it was giving very decent results when we stopped it, see table below).

So we devised a very simple evolutionary algorithm, with accuracy on validation set as fitness function.

[12] ImageNet Large Scale Visual Recognition Competition

Siamese Neural Network - Training results

We tested the resulting networks on the appropriately[13] held-out test set and obtained[14]

Model	Accuracy	Loss
VGG16 + MLP (Adults)	90.43%	0.5991 (H)
VGG16 + MLP (Child)	90.24%	0.5986 (H)
CV (Child)	93.06%	0.5751 (H)
Evo (Child)	**94.07%**	**0.5628** (H)
Evo (Adults)	**93.46%**	**0.5658** (H)

[13] Two different sets, one for adults and one for children

[14] H = Hinge loss

Siamese Neural Network - Selected architectures

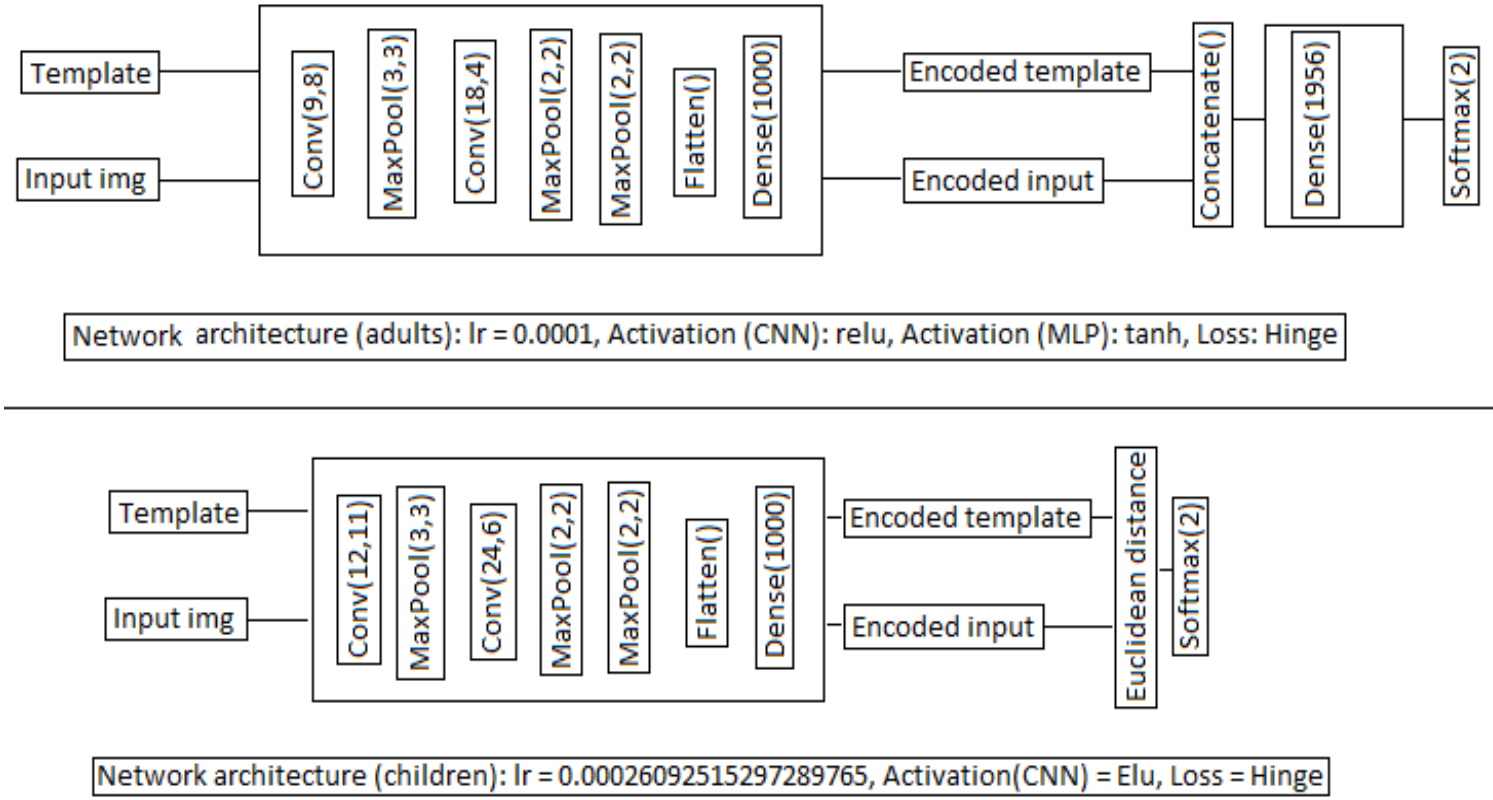


Figure: Selected architectures, Optimizer (both): Nadam

Improvements: Child-Seat Detection System

When used properly, only children should be sitting on the child-seat. This assumption suggests a completely different approach for the detection of the presence of the child in the car.

Instead of relying on the face detection and classification, we can assume that, when a child-seat is detected in a certain region, if a face is detected in the same region then it would belong to a child.

Under this assumption it would be possible to perform the face classification task fewer times in order to increase the general speed of the system.

Improvements: Child-Seat Detection System

We decided to employ SIFT descriptors for determining the region where the child-seat is. These descriptors represent features of the child-seat (owned by the user) which can be use to recognize future instances of it.

The collection of the SIFT descriptors would be performed through the same camera used for the detection. This ensures little variation in the point of view over the child-seat.

Note that the system must include a detector for a generic child-seat since the collection of the descriptors is *bound* to the presence of the child-seat.

Improvements: Child-Seat Detection System

Figure: Detection system block diagram - Overview.

Improvements: Child-Seat Detection System

The steps for our generic child-seat detection system are:

1. Collection of SIFT descriptors from child-seat images;
2. Creation of a Bag Of Visual Words $\mathcal{B}$ using K-Means clustering algorithm;
3. Collection of SIFT descriptors from the camera input stream and representation in the BOVW feature space;
4. We assume that the input $\hat{x}$ contains a child-seat if the following statement holds true for at least n patterns in $\mathcal{B}$

$$\|\hat{x} - x_{\mathcal{B}}\| < \gamma, \quad x_{\mathcal{B}} \in \mathcal{B}.$$

Improvements: Child-Seat Detection System

The following results were obtained by using:

- 197 images taken from the internet;
- 50 clusters when creating the BOVW;
- $n = 5, \gamma = 0.1$.

Input frames

Frames where a child-seat has been detected

Improvements: Child-Seat Detection System

From the frames where a generic child-seat is detected the SIFT descriptors are computed.

Ideally, if the car's backseat is free from clutter, the descriptors should gather around the child-seat. A good way to remove descriptors not belonging to the child-seat (i.e. the outliers) would be to use the mean and standard deviation of the key-points' position.

The detection of the user's child-seat is performed by comparing the SIFT descriptors taken in the previous step (i.e. the second block of the block diagram) with the new ones from the camera input.

Improvements: Child-Seat Detection System

Using two frames (and their horizontally flipped versions) from the unprocessed version of the video below, we obtained the following result.

The red circle is the mean of the key-points (black circles), the green bounding box includes all the key-points while the blue one is a 200×200 pixels square centered in the mean.

RGB vs BGR in OpenCV

While we were testing our final product, we thought about how OpenCV reads images and remembered that it uses the BGR[15] color space.

So we tested how the performance changed by converting the BGR image in RGB and, surprisingly, we observed that MTCNN did detect faces more reliably.

[15] Sembra che lo usino perché era popolare tra i costruttori di videocamere, lo mettiamo? fonte: https://www.learnopencv.com/why-does-opencv-use-bgr-color-format/

RGB vs BGR in OpenCV

Qui ci mettiamo i due video, quello in BGR e quello in RGB, uno accanto all'altro

Thank You.