École Polytechnique Fédérale de Lausanne

Unsupervised deep-learning-based video registration

by Luca Barras

Bachelor Project Report

Approved by the Examining Committee:

Prof. S. Süsstrunk
Thesis Advisor

Hakki Can Karaimer
Thesis Supervisor

EPFL IC IINFCOM IVRL
(Bâtiment BC)
Station 14
CH-1015 Lausanne

April 21, 2020

Abstract

This project aims to align an Infrared (IR) and a RGB video using an unsupervised deep-learning method. It exist already methods to align a pair of IR and RGB image but not again on videos. Because a video is composed of thousands of images, the previous methods was not adapted for as many images. The novelty is that we use an unsupervised deep-learning method to resolve this problem. Receiving a pair of RGB and IR video, we convert the video in frames and we send them to a convolutional neural network with a spatial transform layer. The spatial transform layer allows to transform the frames by applying linear or not linear transformation on the location of the pixels. The output of this network is the IR frame after the transformation that is more aligned with the RGB frame. After to have all the output frames we can reconstruct the video by merging the RGB frames and the new IR frames by applying an alpha value on them. The source code of this project is available online at `https://github.com/lbarras27/deeplearning_registration_IR_RGB_video`.

Contents

Chapter 1

Introduction

1.1 Setting

Nowadays the infrared (IR) is used in several domains, in scientific, military, industrial, law enforcement and medical applications. The most known use is for the night vision. Indeed the infrared cameras capture the heat in contrary to the usual cameras that capture the color that we see. In many applications we want to be able to align an ordinary image (RGB image) with an IR image, but it take works. For example in the medical domain, it is interesting to superpose the two images or the two videos to better see where is the anomalies. If to align an image can be laborious, imagine you to align a video that contain thousands of images in general.

On other side, the deep-learning is becoming more and more used in the scientific domain, mainly in the computer vision domain. For example, it is used for facial recognition or also in the autonomous car. Born in the 80s, the deep learning become really used in the 2010s thanks to the GPU use for this task. The deep learning is probably the most powerful domain in machine learning. The machine learning principle is to train a machine (program) to perform a certain task by giving it a lot of data in input.

1.2 Goals and Motivation

Why do not use the powerful of deep learning to align an IR video with a RGB video ? The alignment of images or videos is called the registration in the jargon of the computer vision engineers. It is that this project aims. We use an unsupervised deep-learning method to achieve this task. Using a supervised method is very laborious because we must label every frames of the video, so unsupervised method seem to be better for this task. There have already been studies on the registration of two IR, RGB images but never again with an unsupervised deep learning method. Recently, there have been a method for the registration of two images with an unsupervised deep-learning network but not with IR images. This project will aims to use this method on IR and RGB videos as dataset. The dataset is a series of IR videos with their corresponding RGB videos. The IR and RGB videos are nearly the same but with a small shift because it is really complicated or even impossible to have exactly the same framing with two different cameras.

Chapter 2

Background

2.1 Infrared light

This is an electromagnetic radiation that have wavelengths longer than those of visible light. So it is invisible to the human eye. For a human the light is visible at a wavelength between 400nm and 700nm but the infrared light has a wavelength between 700nm and 1mm. Almost all heat sources emit infrared light. It is why the infrared camera can see in the night because it capture the heat. We can see on the figure 2.1 that the tea-pot contain a warm liquid for example.

2.2 U-Net Network

The U-Net Network is a special Convolution Neural Network (CNN) that it takes an input and the output has an higher definition than the classical CNN. For example, if we have an image of 300x200 as input, the output can be of same size, 300x200. This network allows to have a higher definition output.
On the figure 2.2 we can see a typical U-Net network architecture. This network is separate in

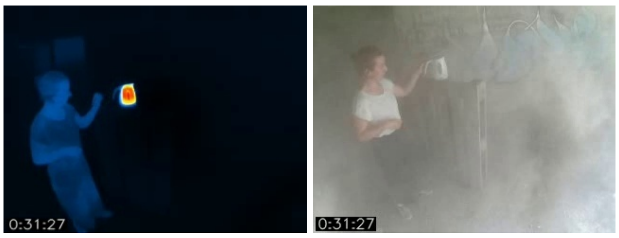

Figure 2.1: Infrared image and ordinary image (RGB)

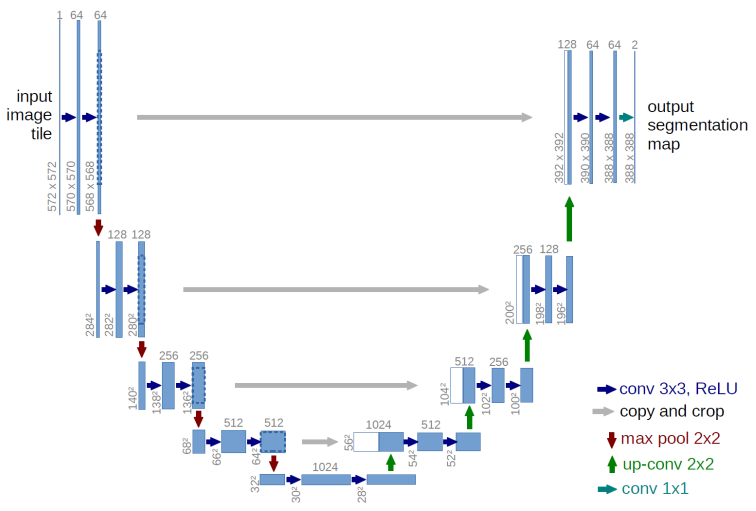

Figure 2.2: U-Net network architecture, figure taken from [**6**]

two parts. The first one is the encoder, in this part the size of the features maps is reduce. The second part is called decoder because the size of the features maps increase. We remark we have connection between each same level where we copy the features maps from the encoder to the decoder. In the encoder part, in general there are a series of convolutions with stride greater than 1 or convolution with pooling (for example max-pooling). In general, we increase the number of channels when we go down and in the decoder part, on the contrary we do convolutions with up-sampling operations and we decrease the number of channels.

2.3 Spatial Transformer Network

A spatial transformer network (STN) is a generalization of differentiable attention to any spatial transformation. It allow a neural network to learn how to perform spatial transformations on the input image in order to enhance the geometric invariance of the model. It can crop, scale, translate and rotate a image for example. It is useful because the convolutional neural networks (CNN) are not invariant to affine transformations in general. Affine transformations are mainly the rotation, scale and translation.
STN is separate in three parts. We can see on the figure 2.3. The first part is the localization network. It is a CNN which regresses the transformation parameters. The second part is the grid generator, it generate a grid that map the position of each pixel of the input image to the corresponding position in the output image. We can see that more precisely on the figure 2.4. And the third part, called the sampler, use the coordinate of the grid to map the pixel positions of the input image to the output image (image after transformation).

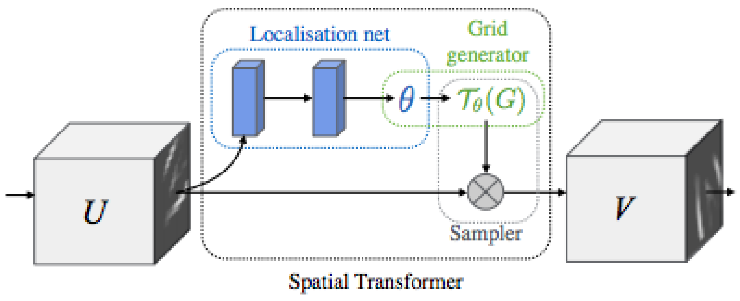

Figure 2.3: Spatial Transformer architecture, figure taken from [**7**]

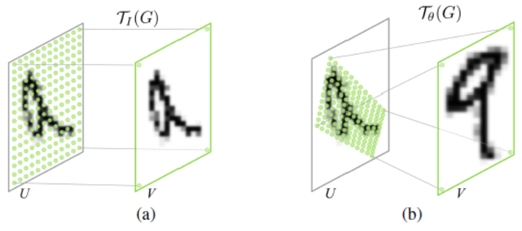

Figure 2.4: In (a), we can see an identity transformation. In (b), we have an affine transformation. U is the initial image and V is the image after the transformation, figure taken from [**7**]

2.4 Voxelmorph

Voxelmorph is a deep-learning based registration library write in python with a Keras and Pytorch version. To achieve to align two 2D or 3D images, it use an unsupervised deep-learning method.

2.4.1 Architecture

The network take two inputs, the fixed image and the moving image. Firstly, it use a U-Net network to get the registration field and then it use a spatial transform on the moving image with the registration field as transformation parameter. The output is the moved image that is the moving image after to have apply the transformation on it. The figure 2.5 show more precisely how is this network.

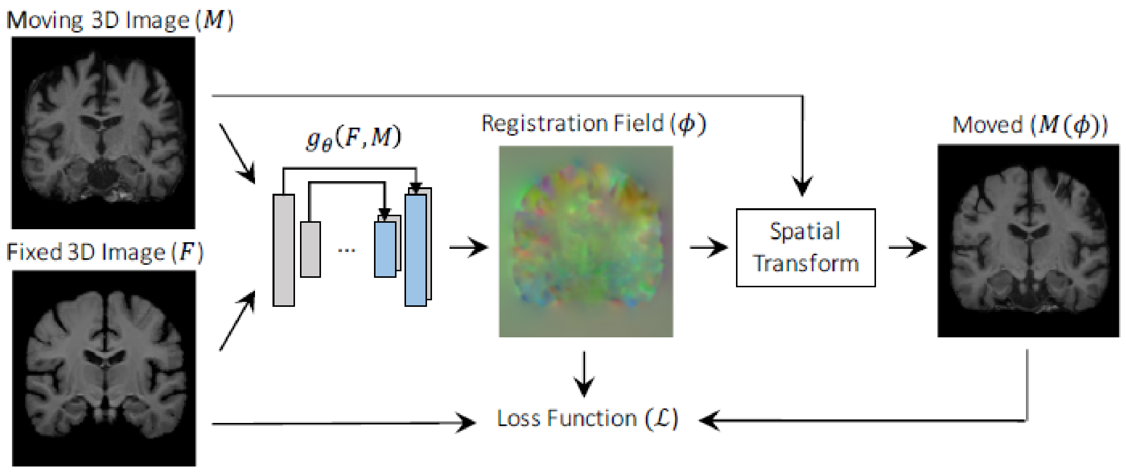

Figure 2.5: Voxelmorph architecture, figure taken from [**4**]

2.4.2 Loss

The loss function of this network is:

$$\mathcal{L}(F, M, \phi) = \mathcal{L}_{sim}(F, M(\phi)) + \lambda\mathcal{L}_{smooth}(\phi)$$

$\mathcal{L}_{sim}(.,.)$ measure the similarity between the fixed input and the moved image. For example $\mathcal{L}_{sim}(.,.)$ can be the mean square error between these two images. $\mathcal{L}_{smooth}(\phi)$ allow the regularization on ϕ. This part impose the smoothness of the transformation. λ is the regularization parameter.

Chapter 3

Method

To achieve the registration between the IR video and the RGB video, we will use two similar methods. First we will use the exactly same method like in Voxelmorph. So we have two inputs in the convolution neural network, the fixed image and the moving image and two output, the flow field and the moved image. In our case, the fixed input is a frame of the RGB video and the moving input is the corresponding frame in the IR video. On the figure 2.5 like explained before in the Voxelmorph architecture section.
For the second method, we use an architecture very similar to the previous one but there is a little difference. We start with a CNN to regress to a 2 x 3 affine matrix. And apply this matrix on the image. So the shape in the image cannot be deform because we apply an affine matrix on the image. In the previous method, we apply a flow field in the spatial transform component. So the pixels can move anywhere, it is for that we add the $\mathcal{L}_{smooth}(\phi)$ part in the loss function. The second method do not need this part in the loss function because the linearity of the transformation is guaranteed by the affine matrix.

3.1 Voxelmorph method

3.1.1 Network architecture

It is the same network used in Voxelmorph. It take as input the concatenation of the RGB frame and the IR frame. So the size is of 2 x 320 x 240 in our case. It begin to enter in the U-Net network. In the first part, it does a series of 2D convolutions with stride equal to 2 followed by a leaky ReLU activation. So the size of the feature maps decrease layer by layer by a factor 2. In the second part, it does 2D convolutions with stride equal to 1 followed by a Leaky ReLU and an up-sampling of factor 2 and a concatenation with the feature map from the encoder which is on the same level. The kernels size of each convolution is 3 x 3. The output of this U-Net network give us the flow field to pass to the spatial transformer. The second input of the spatial transformer is the IR

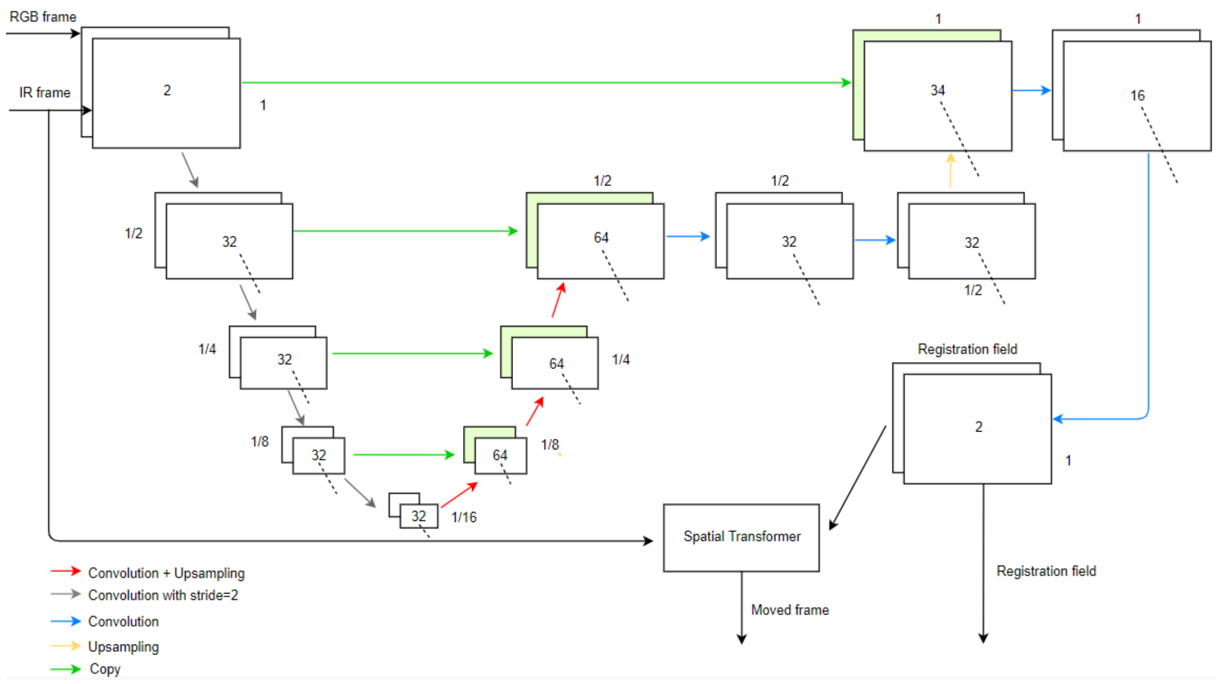

Figure 3.1: The number in center of each box is the number of channels. The number next to or above the box is the spatial resolution with respect to the input. The green boxes are the copied features maps. After each convolution, there is a Leaky ReLU activation.

image (moving image). The function in the spatial transformer is simply the addition of the flow field with an identity grid. So the result grid give us the position to each pixel to map from the original IR frame to the new frame. The output of the spatial transformer is the IR image after the transformation. The two outputs of this network is firstly, the moved image of size 320 x 240 and secondly the flow fields of size 2 x 320 x 240. The network is showed in figure 3.1.

3.1.2 Loss function

The loss function is the same used in voxelmorph. Remember you, it was:

$$\mathcal{L}(F, M, \phi) = \mathcal{L}_{sim}(F, M(\phi)) + \lambda \mathcal{L}_{smooth}(\phi)$$

Where F is the RGB image, M is the IR image and $M(\phi)$ is the network output. $\mathcal{L}_{sim}(F, M(\phi))$ measure the similitude between the IR image and the warped image. In this experiment we use the mean square error as function. $\mathcal{L}_{smooth}(\phi)$ force the smoothness of the transformation. In our case, $\mathcal{L}_{smooth}(\phi) = ||\nabla\phi||^2$. Finally, λ is the regularization parameter, bigger it is, better is the smoothness of the transformation.

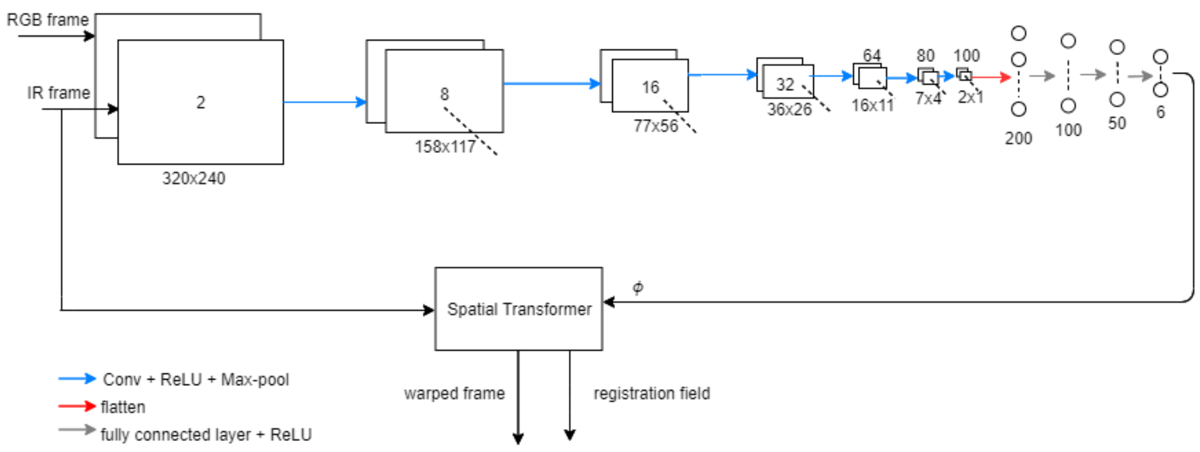

Figure 3.2: The number in center of or above each box is the number of channels and the numbers underneath the boxes are the size of the features maps.

3.2 Affine method

3.2.1 Network architecture

The network is simply a spatial transformer network. It take as input an 2-channel image that is the concatenation of the IR frame and the RGB frame. The input size is of 2 x 320 x 240 in our case. The network begin with a succession of 2D convolutions followed by ReLU activation with kernel size of 7 x 7, 5 x 5, 5 x 5, 5 x 5, 3 x 3, 3 x 3 respectively. All the convolutions are did with a stride of size 1. After we flatten the size to use the fully connection layers. In our experiment, after the flattening, we have a size of 200 x 1. Then the next layers have a size of 100, 50 and 6 respectively. 6 can be seen as 2 x 3 that is the size of the affine matrix. In the spatial transformer we apply the affine matrix on the IR image and get the first output (warped image) of the network. The second output is the flow field generate by the affine matrix. The first output has a size of 320 x 240 and the second a size of 2 x 320 x 240. 2 channels because we have a channel for each direction (x, y). We can see more in details the network architecture on the figure 3.2.

3.2.2 Loss function

The loss function is simply the similitude between the RGB image (fixed input) and the warped image. It can be expressed as:

$$\mathcal{L}(F, M, \phi) = \mathcal{L}_{sim}(F, M(\phi))$$

Where F is the RGB image, M is the IR image and $M(\phi)$ is the IR image after to be went out of the spatial transformer. $\mathcal{L}_{sim}$ is the mean square error in our experiment.

Chapter 4

Experiments

4.1 Dataset

The dataset is a series of IR and RGB video pairs. As our dataset is a series of videos, we must do some preprocessing on it before to can send it in our network. Firstly, we focus on one video pair. The RGB and IR video last 21 minutes each. Then we convert the videos in frames and we obtain 16441 frames for each video. When we have the frames of each video, we must resize the infrared frames because they do not have the same size as the RGB frames. Indeed the IR frames has a size of 320 x 256 pixels and the size of the RGB frames are of 320 x 240 pixels. So we resize the IR frames in 320 x 240 pixels. After that, we remark that the RGB frames contain a lot of very dark images. We remove all these dark frames and the respectively IR frames because we always must have the pair of image to can use in our network. After this operation, we only have 12633 frames. Before to put these data in the network, we must again normalize the pixels value between 0 and 1. Now the data is ready to be send in the network. In the first part of the experience, we just use the gray-scale version of the IR and RGB images. In the second part, we apply some filters to the images pairs. We apply first a Gaussian filter to smooth a little bit the images. Then we apply a Sobel filter in each direction (x, y) and compute its magnitude: $\sqrt{(sobelx)^2 + (sobely)^2}$. We get the contours of the images. We can see that on figure 4.2.

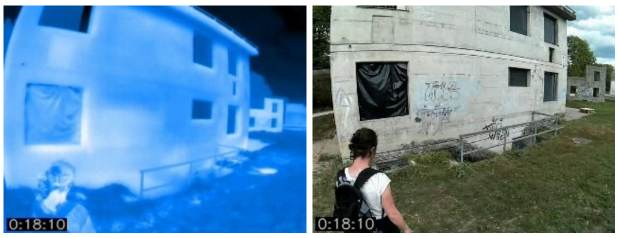

Figure 4.1: Example of a pair of IR and RGB frame

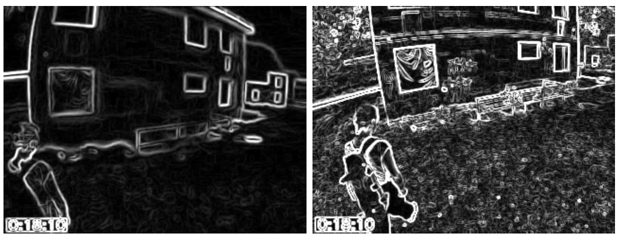

Figure 4.2: Example of a pair of IR and RGB frame after to have applying filters

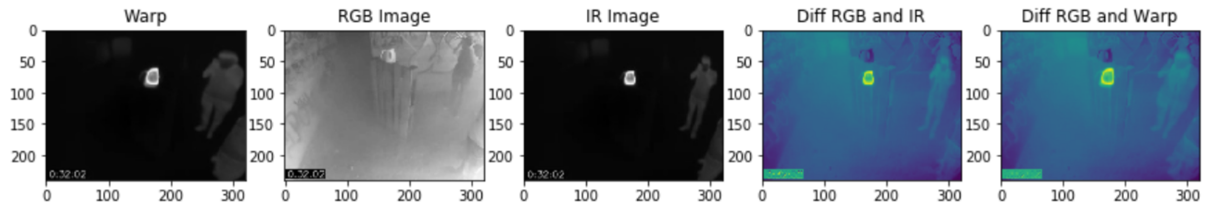

Figure 4.3: Example of a result of the print function

4.2 Implementation

To manipulate the dataset we use OpenCV and Numpy libraries. The networks are coded using Pytorch. For the first method (Voxelmorph method), we use the Pytorch version of the network implemented in the Voxelmorph paper but with little modifications. Indeed, in their Pytorch version, sometimes the code is not adapted to work with 2D images but only for 3D images. So we rewrite their network to make it work with 2D images. For the two networks, we use ADAM optimizer with a learning rate of 0.0001. To speed up the learning, we use a batch of 32 pairs of images for the first network and for the second network we use a batch of 100 pairs of images. To better see if an IR frame is aligned with its corresponding RGB frame, we implemented a print function that show the IR image, RGB image, warped image, the difference between the RGB and IR image and the difference between the RGB and the warped image. Figure 4.3 show us a typical print of this function. The second tool we implemented to see if the frames are aligned is to reconstruct the video after to have merged the warped frame (IR frame after the transformation) and RGB frame. To merge the two images, we apply an alpha blending on them. Alpha blending is just an interpolation between the two images. The formula is given by: $\alpha M(\phi) + (1 - \alpha)F$, where F is the RGB frame and $M(\phi)$ is the warped frame. Figure 4.4 show the result of the merging of the two images with $\alpha = 0.35$. The source code of this project is available online at `https://github.com/lbarras27/deeplearning_registration_IR_RGB_video`.

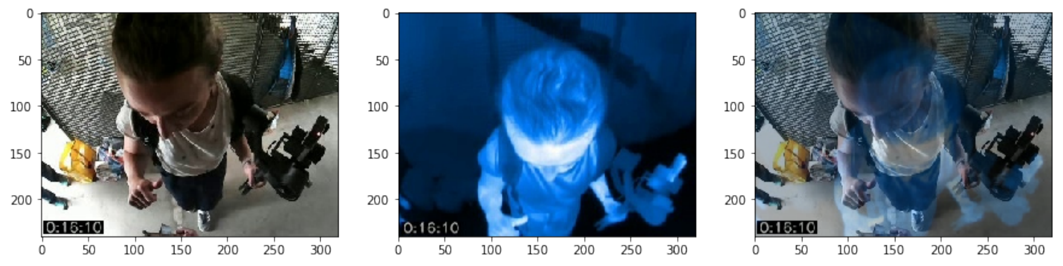

Figure 4.4: We can see the result of the merge of the RGB and IR frame with $\alpha = 0.35$. RGB, IR and the fusion frame respectively.

4.3 Results

The results are not very good, we can see on the figure 4.3. We did 20 epoch for each experiment. We tried to do more epoch but the results getting worse and worse. We varied λ between 0.1 to 0.6 for the first method and we have the output image too warped when we use a small value for λ and not aligned. And when we use a bigger value the result is not much warped but not aligned too. With the second method, the output is too scale and not at all aligned with the RGB frame. For the second part of the experience, when we use the frames after to have applied some filters at the input of the networks, the results are not what we want too.

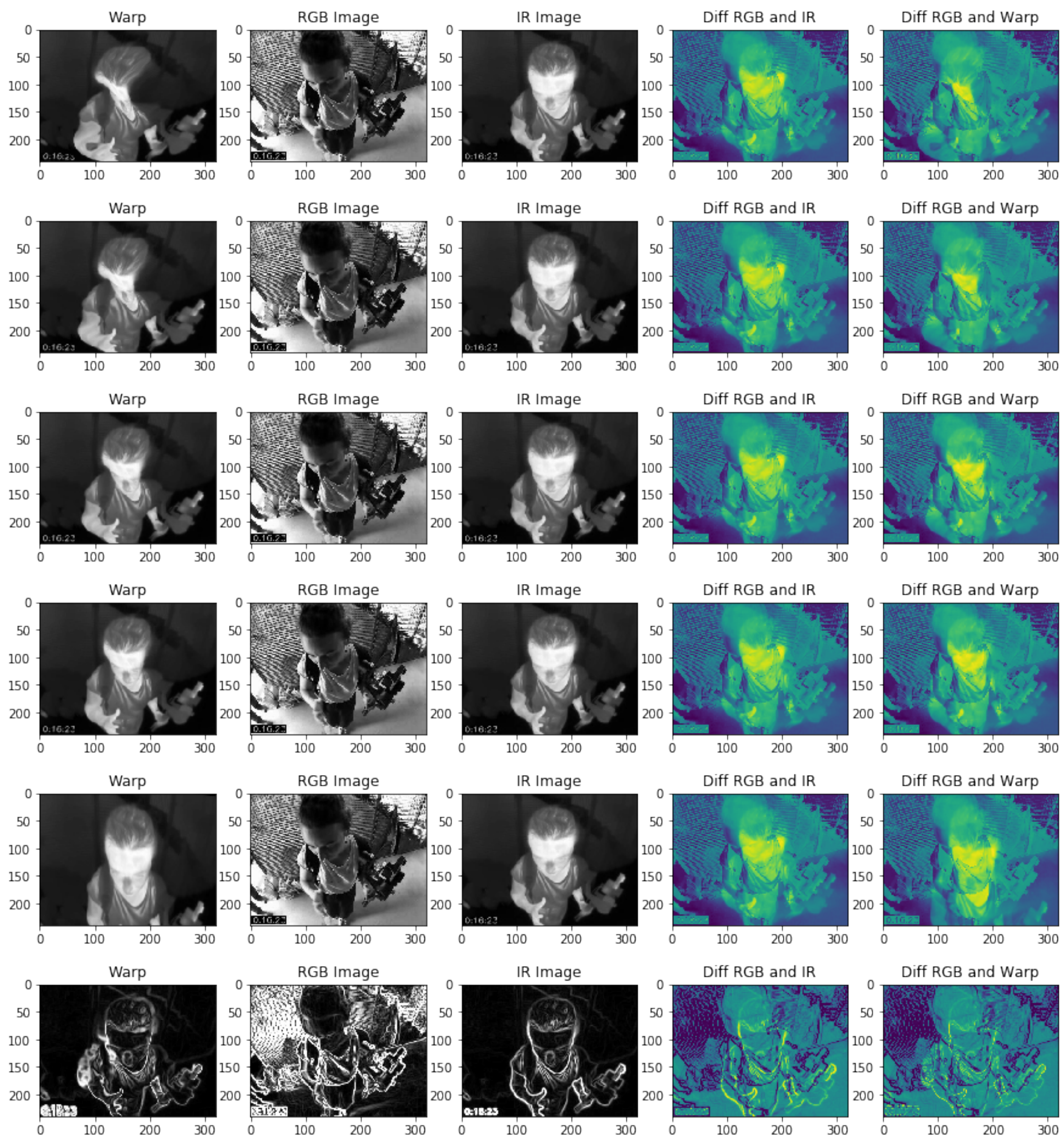

Figure 4.5: Results for the first model with $\lambda = 0.1$, $\lambda = 0.2$, $\lambda = 0.4$, $\lambda = 0.6$. Then the second model and finally with filter frames with the first model with $\lambda = 0.2$ respectively.

Chapter 5

Discussion

In the first part of the experience the results are bad because we cannot really compare the RGB frame with the IR frame because the pixels value are not correlated between them. For example, if on the RGB image a person has a black t-shirt or a white t-shirt, on the IR image it will appear like it was the same ones. We can see that on figure 5.1. So if we do the mean square error between the IR and RGB images, it is going to measure the similitude between the two images that is not what we want. The only correlation between these two images, it is the shape borders in the image.
It is why in the second part of the experience we used the frames that contain the contours of the shapes. But we had not good results too. We think it is because there is much more contrast on the RGB frames. We can see it on figure 5.2.

5.1 Future work

So for the rest of the semester, we will focus to improve this technique in lowering the contrast of the RGB image.

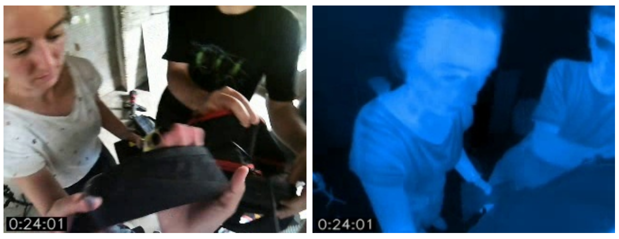

Figure 5.1: We can see that the t-shirt white (very light) and the t-shirt black (very dark) on the RGB frame appear the same color on the IR frame

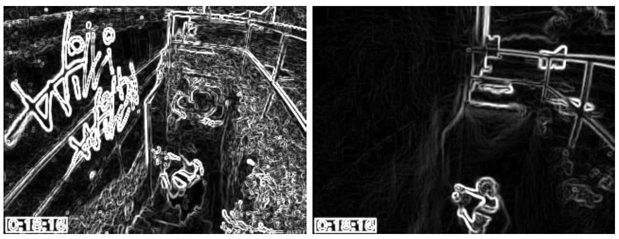
Figure 5.2: We remark there is so much contrast on the RGB image (on the left)

An other point, on the place to filter the frames before to send them in the network, we can try to filter them in the loss function to not lose some information about the initial frames. Then we are going to try to take account the direction of each border to improve the loss function.

Chapter 6

Conclusion

As we could see, the registration between a IR and RGB image is not as obvious. We cannot just compare the similitude between the pixel values but we must find a correlation between the RGB and IR images. Without to use a correlation, an unsupervised deep-learning algorithm is not effective. The only correlation we can observe is the borders of the shapes in the image. So we applied some filters (Sobel and Gaussian) to bring out these borders, but the result is not good yet. We have a lot more contrast on the RGB image than the IR image. So during the next weeks of this semester, we will focus on that and try to improve the loss function by taking into account the direction of the contours of the shapes in the frames.

Bibliography

[1] Unsupervised Learning of Probabilistic Diffeomorphic Registration for Images and Surfaces Adrian V. Dalca, Guha Balakrishnan, John Guttag, Mert R. Sabuncu MedIA: Medial Image Analysis. 2019. eprint arXiv:1903.03545

[2] https://en.wikipedia.org/wiki/Infrared

[3] Unsupervised Learning for Fast Probabilistic Diffeomorphic Registration Adrian V. Dalca, Guha Balakrishnan, John Guttag, Mert R. Sabuncu MICCAI 2018. eprint arXiv:1805.04605

[4] VoxelMorph: A Learning Framework for Deformable Medical Image Registration Guha Balakrishnan, Amy Zhao, Mert R. Sabuncu, John Guttag, Adrian V. Dalca IEEE TMI: Transactions on Medical Imaging. 2019. eprint arXiv:1809.05231

[5] An Unsupervised Learning Model for Deformable Medical Image Registration Guha Balakrishnan, Amy Zhao, Mert R. Sabuncu, John Guttag, Adrian V. Dalca CVPR 2018. eprint arXiv:1802.02604

[6] U-Net: Convolutional Networks for Biomedical Image Segmentation Olaf Ronneberger, Philipp Fischer, Thomas Brox. eprint arXiv:1505.04597

[7] Spatial Transformer Networks Max Jaderberg, Karen Simonyan, Andrew Zisserman, Koray Kavukcuoglu. eprint arXiv:1506.02025