Siamese CNNs for RGB-LWIR Disparity Estimation

David-Alexandre Beaupre Guillaume-Alexandre Bilodeau
LITIV lab., Department of Computer and Software Engineering, Polytechnique Montreal
{david-alexandre.beaupre, gabilodeau}@polymtl.ca

Abstract

Currently, for the task of color (RGB) and thermal infrared (LWIR) disparity estimation, handcrafted feature descriptors such as mutual information are the methods achieving best performance. In this work, we aim to assess if convolutional neural networks (CNNs) can achieve competitive performance in this task. We developed an architecture made of two subnetworks, each consisting of the same siamese network, but taking different image patches as input. Each siamese network, in the feature space, searches for the disparity between the left and right patch. The output of the two subnetworks are summed together so that we can be more confident in the predicted disparity by enforcing left-right consistency. We show that having two subnetworks working together in parallel to get the final prediction helps achieve better performance when compared to a single subnetwork by itself. We tested our method on the LITIV dataset and found the results competitive when compared to handcrafted feature descriptors. The source code of our method will be available online[1] *upon publication.*

1. Introduction

Stereo matching remains one of the most fundamental task in the field of computer vision. With it, we can register stereo images from two different cameras into the same coordinate system from which operations such as depth estimation and object detection can be performed. Recently, more work with stereo image pairs outside of the visible domain has been done. Having one of the images in the LWIR (Long Wavelength Infrared or thermal infrared) spectrum can help with some problems affecting a stereo image pair in the visible spectrum. For example, a person wearing dark clothes at night will be more difficult to perceive in the visible spectrum than in the infrared spectrum. Anytime a person has a low color contrast with its environment, detection in the visible domain will be challenging. However, in the thermal infrared domain, this will be much easier because we use the heat of the person's body to do the detection. In the LWIR domain, detections do not rely on textures or colors, but simply on the person having a different temperature from its environment. Nonetheless, this restricts the class of objects that we can work with *i.e.* we cannot work with objects that do not emit heat. For example, stereo matching is a method widely used in self-driving car and the usage of an infrared camera, in this case, would not be always helpful as road signs and urban furniture often do not emit heat. This is because objects of interest need to have a different temperature than the ambient temperature in order to be detected. So, in the case of humans, working in the infrared domain makes sense because our body temperature, in most cases, is warmer than the ambient temperature when working with indoor or outdoor videos. Thermal images, while not perfect, seem well adapted for detecting humans and their combination with visible images can be beneficial. This is why the proposed model is targeted to do disparity estimation for human silhouettes.

Recently, convolutional neural networks (CNNs) have shown to perform very well in multiple tasks of computer vision such as object detection [18, 19], classification [13, 24, 9], tracking [28, 29] and stereo matching [5, 11, 12, 17, 15, 30, 20, 6]. Most of the recent works in deep learning for stereo matching is, however, mostly in the visible domain, where we know that CNNs are able to extract meaningful features in order to do matching. In the thermal infrared domain, there is less textures and intensity differences, therefore the task of stereo matching between the visible and thermal domain is harder [2]. This work aims at investigating if the recent successes of deep learning in visible stereo matching can be translated to visible-thermal stereo matching. More precisely, we want to assess if a method that learns descriptors outperform the ones using handcrafted feature descriptors that are presently the state-of-the-art for visible-infrared image pairs. Classical methods to match pixels from two different spectrum (visible and thermal) often consist of using descriptors (SIFT [14], HOG [7], mutual information [27] and others) and then using a sliding a window approach to match the features between the images.

[1] Available at https://github.com/beaupreda

We developed a new model, consisting of two subnetworks working in parallel, where each subnetwork receives a pair of multispectral stereo image patches as input. The domains of these inputs are inverted for each subnetwork. Each input pair is made of a small square patch and a larger patch. The stereo pair image patches are forwarded into a CNN module (siamese network) where the goal is to find the location of the small patch inside the bigger one. This is done through a siamese network made of convolutions, batch normalization [10] and ReLU activations [16], with the two branches being merged with a correlation layer to output a log-probability vector for each possible disparity location. Since we have two subnetworks working in parallel, we have two vectors of log-probabilities (one that predicts the disparity of RGB inside LWIR, and the other that predicts LWIR inside RGB). We sum those two log-probabilities vector and take the index of the maximum value to get our final disparity prediction. The goal of having two subnetworks is that they enforce some sort of consistency by learning to give the same disparity prediction. In summary, our main contributions are:

- We propose a new model made of two subnetworks, each of them taking multispectral stereo pair image patches as input. The outputs are log-probabilities vector which when summed, give a disparity prediction between the visible and the infrared domain. Our model also enforces left-right consistency for the disparity predictions.

- We perform experiments on the LITIV dataset [3] to validate the performance of our model. These results show that a CNN is able to be competitive with handcrafted feature descriptors.

The organization of the paper is the following: in section 2, we discuss related work. In section 3, we present our proposed model. In section 4, we present the dataset used to evaluate our method, some data augmentation that we did and our results. Finally, in section 5, we conclude this paper.

2. Related Work

In the task of disparity estimation for human silhouettes between the visible and the thermal infrared domain, handcrafted feature descriptors are still the preferred approach. There are three families for categorizing similarity measures. The first category consists of methods that compute similarity across all the pixels inside a given window. Methods such as mutual information [27] fall into this category. Mutual information [27] computes statistics of cooccurrence of intensities for all pixels of two given windows. Because it is able to find a match between a textured region and an uniform one, this method gives good result for disparity estimation between two different domains. Another method falling into this category would be Sum of Squared Differences (SSD) [3] consisting of summing the squared difference of each pixels in two windows, one in each image of an image pair. The second category includes methods that model data as distributions. This category incorporates methods [3, 25, 26] that rely on descriptors such as LSS [21], SIFT [14] and HOG [7]. LSS is a local descriptor able to capture self-similarity among colors, textures and repetitive patterns, which makes it, like mutual information, proficient at matching textured regions with uniformed ones. SIFT is different from LSS because it is based on gradients. SIFT uses interest points that are invariant to illumination, rotation, viewpoint and scale to match objects from two different images or windows. For HOG, since it is already based on windows, we simply need to measure the distance between two histograms coming from two different windows to get a similarity score. The third category measures the binary comparison of pixels and include methods [3] based on FREAK [1] and BRIEF [4]. Both of these features compute a binary vector representing each window to match and with the hamming distance, the similarity between the windows is obtained. These methods have the advantage of being faster than SIFT.

In the task of stereo matching in the visible domain, deep networks are the methods that achieve state-of-the-art results. We can separate deep learning methods into two categories: the ones that are patch-based and the ones doing end-to-end learning. In the patch based methods, *Zbontar et al*. [30] were the first to show that it was possible to use a CNN to do stereo matching. Their method consisted of taking two small patches from a stereo image pair and with a CNN, classifying if the two patches were a good match or a bad match. *Luo et al*. [15] takes one small patch from the left image and a larger one from the right image and treats the problem as a multi-class classification, where the different classes are all possible disparity values. They also join the features of their siamese network with a inner product which produces very good results in term of computation. Our siamese network showed in section 3.2 is inspired by this previous work. *Jie et al*. [11] used constant highway networks [20] to produce a stereo matching cost volume and then used it as input inside a recurrent neural network (RNN) to do the disparity estimation. The main idea was to learn the left-right consistency verification during training and utilizing the error maps given by that verification as attention module to guide the network towards those areas during training. Constant highway networks [20], as mentioned above, produce a matching cost volume based on both inner and outer residual shortcuts. They also introduce a way to output a disparity map and a confidence of said map with the creation of a global disparity network featuring a reflective confidence. For the end-to-end methods,

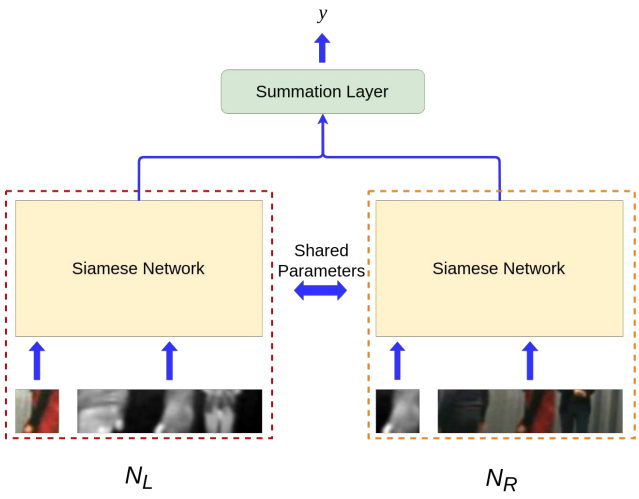


Figure 1. Architecture of our proposed model showcasing the two subnetworks N_L and N_R.

Kendall et al. [12] said that many problems in stereo matching could be solved by using geometry *i.e.* knowledge of the environment around the matching points. They proposed using 3-D convolutions to incorporate context and act as a regularizer over the cost volume. They also show that treating the problem of stereo matching as regression gives better performance than treating it as a classification problem. *Chang et al.* [5] also tries to take advantage from context information. In order to do this, they introduced a spatial pyramid pooling module to get different scales *i.e.* generate feature maps of different sizes and also introduced a stacked hourglass (encoder-decoder architecture) to get more information from the context. *Pang et al.* [17] proposed an end-to-end cascade method composed of two stages. The first stage is responsible of producing fine-grained disparities with an up-convolution module and is the input of the second stage where the disparities are rectified with residual signals.

Our method is based on patches because densely annotated visible-thermal infrared datasets are not available.

3. Method

This section presents the proposed model made of two subnetworks working in parallel, each being a siamese network, to achieve the final disparity prediction. The global architecture can be viewed in figure 1.

3.1. Network Architecture

Our network is composed of two siamese networks taking a total of four different inputs to achieve a final disparity prediction y for a given location. We will refer to each subnetwork (siamese network branch) as N_L and N_R for the left and right subnetworks respectively. There are two inputs for each subnetwork, the left input being a small square patch of size $p_s \times p_s$ and the right input being much larger, but of the same height, of size $p_r \times p_s$ where p_s and p_r are defined as follows:

$$p_s = 2 \times p_{hs} + 1 \tag{1}$$

$$p_r = 2 \times p_{hr} + p_s + 1 \tag{2}$$

where p_{hs} and p_{hr} are hyperparameters of the network, where they represent half the size and range of a patch. These two values will be discussed in section 4.1. The subnetwork N_L has two patches as inputs, $P_{N_L}^{RGB}$ and $P_{N_L}^{LWIR}$, the first one being in the visible domain (RGB) and the other one being from the infrared domain (LWIR). For the N_L branch, the patch $P_{N_L}^{RGB}$ is the left input while $P_{N_L}^{LWIR}$ is the right input. For the other subnetwork (N_R), the domains of the inputs are simply inverted, so the left input is $P_{N_R}^{LWIR}$ and the right one is $P_{N_R}^{RGB}$. Basically, we try to find the location of an RGB patch inside an LWIR patch in subnetwork N_L, while in N_R, we try to find the location of an LWIR patch inside an RGB patch. N_L and N_R share all their parameters.

Each subnetwork, composed of a siamese network module (see figure 2, outputs a vector of log-probabilities, $\log(p_L)$ and $\log(p_R)$ for the left and right branches respectively, of size p_r. These vectors represent the probability of finding the center of the left patch at disparity location d inside the larger right patch. We can then define the prediction of the left and right branch as y_{N_L} and y_{N_R} respectively. Instead of working with only one of the two branches and taking the maximum of either p_L or p_R as the prediction, we sum those two vectors together and take the position j of the maximal value as the prediction for the disparity. Formally we can then define the final prediction y as follows:

$$y = \arg\max_j(\log(p_L) + \log(p_R)) \tag{3}$$

This is done for a number of reasons. First of all, adding the predictions of both subnetworks N_L and N_R enforces some sort of left-right consistency where the two subnetworks will work together to learn to give the same disparity prediction y_L and y_R. If both branches consistently give the same output, then we can be more confident in the said prediction. Second of all, there are cases where one of the two subnetworks give more information than the other one. For instance, let us say that the distribution of p_L is mostly uniform and therefore, there are no clear disparity location where a maximum arises. If we were to only use this branch for our final prediction y, we would obtain incorrect predictions most of the time. However, in a case like this one, the distribution of p_R might have a clear maximum at a correct disparity location, meaning that if we add both log-probability vectors together, p_L being mostly uniform and

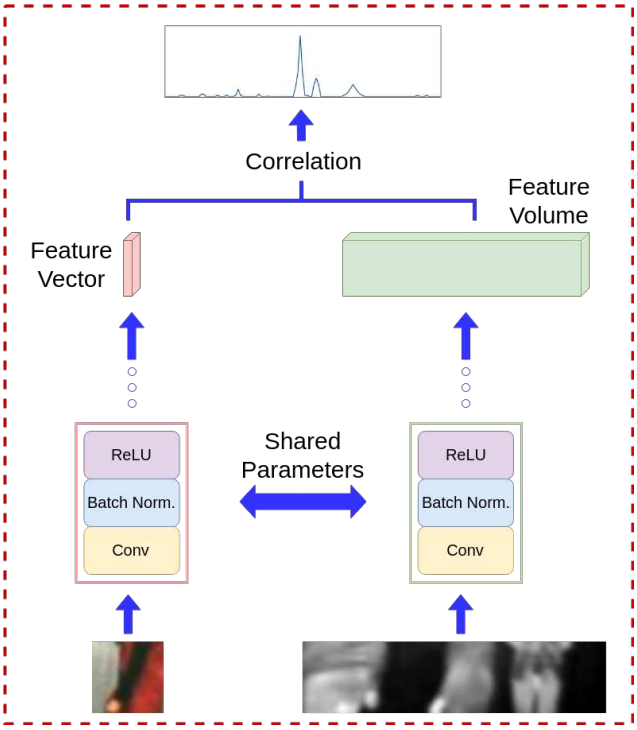


Figure 2. Details of the siamese network (N_L)

p_R having a clear maximum, we will get a disparity prediction that is informative. So, in the cases where finding the best match is easier in either N_L or N_R, having the two branches makes our model much more consistent in its predictions. This will also be shown in section 4.3 where we compare our model with the individual subnetworks and show that there can be variation in performance between N_L and N_R.

3.2. Siamese Network

This module is the same for both subnetworks N_L and N_R. As stated in section 3.1, the only difference is the domains of the input patches, being either RGB-LWIR or LWIR-RGB. In this section, we will use the left branch (N_L) to showcase the inner workings of the CNN module which is shown in figure 2. This module is a siamese network that takes a pair of stereo images as inputs $P_{N_L}^{RGB}$ and $P_{N_L}^{LWIR}$ and is inspired by [15]. Each patch is forwarded into a network of six layers composed of convolutions, batch normalization [10] and ReLU [16]. There is no ReLU activation for the last layer. For the convolutional layers, we used 32 filters of size 7×7, for the first two layers, and 64 filters of the same size for the remaining four layers. We also applied Dropout [22] to reduce overfitting because the datasets that we were working with are somewhat small.

After the left patch $P_{N_L}^{RGB}$ exits the sixth and last layer, we are left with a 64 dimensional feature vector $\vec{v}$. For the right patch $P_{N_L}^{LWIR}$, we have a feature volume of size $p_r \times 64$, named $\boldsymbol{M}$. We then do a correlation operation to obtain a vector of scores $\vec{s}$ of size p_r, with one score for each possible disparity location. We compute $\vec{s}$ as follows:

$$\vec{s} = \vec{v}\boldsymbol{M} \tag{4}$$

This means that each element s_i of $\vec{s}$ is the result of the dot-product between the feature vector $\vec{v}$ from the left patch $P_{N_L}^{RGB}$ and the column i of the feature volume $\boldsymbol{M}$ given by the right patch $P_{N_L}^{LWIR}$. Once we have our vector of scores, we feed it into a *LogSoftMax* layer to obtain the log-probability vector $\log(p_L)$ that is the output of one the CNN module.

As stated before, the same process is done in the network N_R with the left input $P_{N_R}^{LWIR}$ and the right input $P_{N_R}^{RGB}$ to get $\log(p_R)$.

3.3. Training

During training, we extract the small patches $P_{N_L}^{RGB}$ and $P_{N_R}^{LWIR}$ at pixel locations (x, y) for which we know the true disparity d. We also take the larger patches as $P_{N_L}^{LWIR}$ and $P_{N_R}^{RGB}$ at pixel locations $(x+d, y)$. These pair of patches are then forwarded into our network as presented in section 3.2. We use the Adagrad [8] optimizer during backpropagation.

For our loss function, we minimize the cross-entropy with regards to the weights w just as [15] did. Our loss function is defined as follows:

$$loss = \min_{\boldsymbol{w}} \sum\nolimits_{i,y_i} p_{gt}(y_i) \log p_i(y_i, \boldsymbol{w}) \tag{5}$$

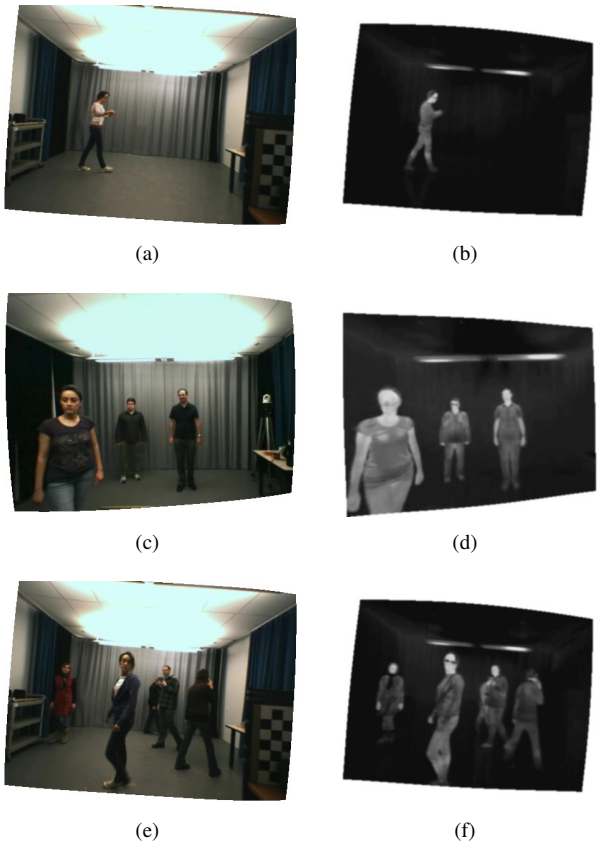
We also smooth the target distribution centered around the value of the ground-truth disparity. Since we are interested in the 3-pixel error metric, the possible values for the target distribution are defined as:

$$p_{gt}(y_i) = \begin{cases} \lambda_1, & \text{if } |y_i - y_i^{GT}| = 0 \\ \lambda_2, & \text{if } |y_i - y_i^{GT}| = 1 \\ \lambda_3, & \text{if } |y_i - y_i^{GT}| = 2 \\ \lambda_4, & \text{if } |y_i - y_i^{GT}| = 3 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

In this work, we set these values as $\lambda_1 = 0.32$, $\lambda_2 = 0.40$, $\lambda_3 = 0.20$ and $\lambda_4 = 0.08$. With this target distribution, we can be sure that the network learns to minimize the disparity prediction error.

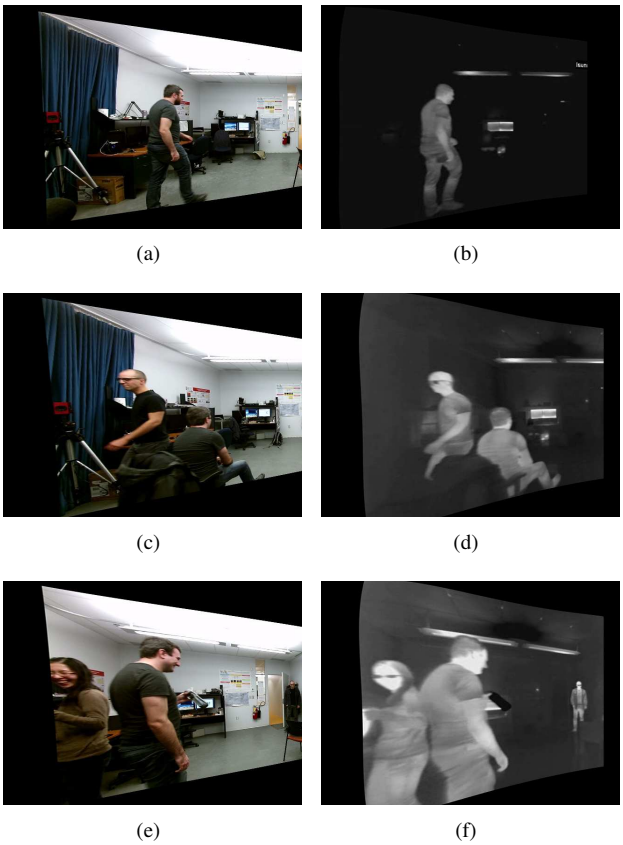
4. Experiments

This section will present the datasets used to conduct our experiments and present the various hyperparameters used for our model. We will also discuss the different manipulations that we did on our data in order to augment the



(a) (b) (c) (d) (e) (f)

Figure 3. Some examples of the multispectral stereo image pairs found in the LITIV dataset, one row being a different video sequence. First column is in the RGB domain while the second column is in the LWIR domain.



(a) (b) (c) (d) (e) (f)

Figure 4. Example of a multispectral stereo image pair found in the St-Charles *et al*. dataset. Each row corresponds to a different video sequence featuring from one to three subjects in each of them.

original datasets. Finally, there will be a discussion about the obtained results.

4.1. Experiment Details

During training, we used two different datasets, one being the LITIV dataset [3] and the other one being the St-Charles *et al*. [23]. All images, in both datasets, were rectified so the search for the matching point will be in one dimension *i.e*. at the same y coordinate in the images of the stereo pair. The LITIV dataset, showed in figure 3 contains three video sequences, named vid01, vid02 and vid03, each of them consisting of people walking in a room at different depths. The number of person in a scene varies between one and five. The main difficulty in these videos are the occlusions between the different subjects and the visual appearance between the modalities. The video sequences respectively contain 89, 67 and 53 annotated images. Disparities are annotated for 11 166 points in vid01, 7529 points in vid02 and 6524 points in vid03. The St-Charles *et al*. dataset, showcased in figure 4 is also separated in three video sequences (v04, v07 and v08) and consists of one to three subjects walking in a room at different depths. Once again, the main difficulty is the amount of occlusion between the people walking in the room and the visual appearance between the modalities. The first sequence, v04, contains 117 images and 4252 disparity points, the second sequence, v07, has 144 images and 5653 disparity points and the last sequence, v08, contains 89 images for 5277 disparity points.

Our model was compared against handcrafted feature descriptors that required no training and that were tested on the LITIV dataset [3], so we had to separate the dataset in three different folds in order to do a fair comparison with the other handcrafted feature methods and we applied a three-fold cross-validation. Table 1 summarizes the separation for each fold. The St-Charles *et al*. dataset was always part of the training data exclusively, while the LITIV dataset was separated between training, validation and testing. One important thing to notice is that the amount of training and testing points differ greatly depending on which video sequence we use, while the number of validation points always stay the same. We had to use exactly 10 000 validation points because we were limited by the memory of our GPU.

	LITIV dataset			St-Charles dataset
	Training	Evaluation	Testing	Training
Fold 1	96 516 (vid01, vid02)	10 000 (vid01, vid02)	32 106 (vid03)	65 310 (vid04, vid07, vid08)
Fold 2	67 608 (vid02, vid03)	10 000 (vid02, vid03)	60 978 (vid01)	65 310 (vid04, vid07, vid08)
Fold 3	80 622 (vid01, vid03)	10 000 (vid01, vid03)	45 678 (vid02)	65 310 (vid04, vid07, vid08)

Table 1. Number of points used for training, evaluating and testing for both the LITIV and St-Charles *et al*. dataset. These number of points are after the data augmentation of section 4.2. The three different folds result in the testing of our method over all the images in the dataset.

The fold 1 is trained on all the St-Charles *et al*. data plus the images from vid01 and vid02, minus 10 000 randomly selected images kept for the validation. Vid03 was kept for the testing. For fold 2, the training set was composed of all the image pairs of the St-Charles *et al*. dataset with the addition of vid02 and vid03, once again, minus 10 000 randomly chose image pairs kept for the validation set. We tested on all the images in vid01. For the last fold, the training data was taken from the entire St-Charles *et al*. dataset in combination of vid01 and vid03 of the LITIV dataset, excluding the 10 000 image pairs that were put aside for the validation. The testing was made on the video sequence vid02 of the LITIV dataset. With these three folds, we are able to test our model on all three video sequences vid01, vid02, vid03. With a weighted average, we will be able to compare our method with the handcrafted feature descriptors that were all tested on the entirety of the LITIV dataset.

For all three folds, we kept the same hyperparameters. We will list the ones that are the most important *i.e*. the ones that have an impact on the results. The first ones are p_{hs} and p_{hr}, corresponding, respectively, to half the size and half the range of one patch. They are set to 18 and 60 respectively. As stated in section 3.1, these are the two parameters that control the size of the inputs of the CNN module. Dropout was also applied on our model. We used a drop-rate of 0.5 for all six layers, except the input layer which has a drop-rate of 0.2 as was suggested in the paper [22]. The learning rate α was set to 0.001 and was decreased after 24 000 iterations by a factor of $\frac{\alpha}{5}$. After that, every 4000 iterations, the learning was further decreased by the same rate. We trained our network for a total of 40 000 iterations, which correspond to 20 epochs. The batch size was set to 64.

4.2. Data Augmentation

One of the difficulties of working with stereo image pairs of both visible and infrared domains is that it can be quite hard to come by large datasets. We worked with both the LITIV dataset and the St-Charles *et al*. dataset and together, they contained a little more than 40 000 points. However, since we worked with patches, some of these points that were close to the images border were not valid, so in reality, we had less points to train our model. In order to have more data to work with, we did some data augmentation operations. The first manipulation we did was to mirror every image in both dataset, which simply doubles the amount of data available. An example of a mirrored RGB-LWIR stereo image pair is shown in figure 5. The other operation applied on the data is that for each ground-truth point $p = (x, y)$ which we know the disparity d, we created two neighboring points one pixel above and one pixel below. Formally, we created the point $p_a = (x, y + 1)$ and $p_b = (x, y - 1)$ with the same disparity d (we are assuming that neighboring points should most of the time have the same disparity). This has the effect to triple every point, so with these two data augmentation operations, we have multiplied by a factor of six the amount of data that we can work with. Concretely, we now have a total of 136 300 points for the LITIV dataset and 65 310 points for the St-Charles dataset. Although we could benefit of having more diversified data, the quantity that we now have is sufficient to train and test our model effectively.

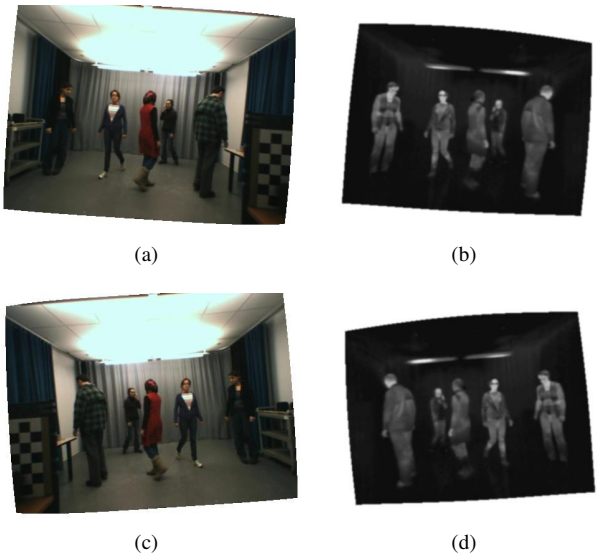


Figure 5. Example of the mirror operation of the data augmentation. The first row shows the original multispectral stereo image pair while the second row shows the mirrored image pair.

4.3. Results

We used the recall metric to do the comparison between our model and the handcrafted feature descriptors. In this

	Recall		
	Proposed model	Left subnetwork (N_L)	Right subnetwork (N_R)
Fold 1	**0.839**	0.753	0.705
Fold 2	**0.783**	0.663	0.642
Fold 3	**0.731**	0.555	0.614

Table 2. Ablation study results of our model with three network configurations: with both subnetworks N_L and N_R, and with each of them individually. **Boldface**: best results.

Method	Recall
Proposed model (37×37)	*0.779*
Mutual Information (40×130)	**0.833**
Mutual Information (20×130)	0.775
Mutual Information (10×130)	0.649
Fast Retina Keypoint (40×130)	0.641
Local Self-Similarity (40×130)	0.734
Sum of Squared Differences (40×130)	0.656

Table 3. Results of our model, average over the three folds, compared with best handcrafted feature descriptors as reported in [3]. Patch sizes are in parentheses. **Boldface**: best result, *italic*: second best.

case, the recall is defined as the correct number of matches on the total number of matches. Formally, using the 3-pixel error, we can define the recall as:

$$recall = \frac{m_{correct}}{m_{total}} \tag{7}$$

where $m_{correct}$ is the number of correct matches and m_{total} is the total number of possible matches. A match is considered correct if the predicted disparity y is at exactly three pixels or less than the ground-truth disparity.

Table 2 shows the results of our method compared with the performance of each individual subnetwork, where the prediction of either N_L or N_R is considered as the final disparity prediction y. We evaluated these three networks on all three folds that were presented in table 1. As we can see, this ablation study shows that the results of our proposed model are always better than the individual subnetworks. For the first fold, our model with the two subnetworks achieves a better recall by a margin of 8% when compared to the best individual subnetwork. The margin is even bigger for the second and third folds, being of 12% and 11% respectively. Therefore, it demonstrates that the use of two parallel subnetworks help improve the performance of our model. This is expected because, as we explained earlier in section 3.1, when working with the two branches, our model can learn to have a similar prediction as the output of both subnetworks N_L and N_R which gives us a high confidence in the disparity prediction by verifying the left-right consistency. When working with only one branch (either N_L or N_R), whatever the prediction, we have to take it as final, which will lead to more errors than the case where we have the predictions of both branches.

Table 3 compares the handcrafted feature descriptors as reported in [3] with our proposed method. We took as a comparison, the best results reported. Therefore, the patches for the handcrafted features include more pixels as the patches have the same width as us, but are about three times higher. The recall for our network is a weighted average with the weights being proportional with the number of evaluated points from table 1. This way, we can have a fair comparison between our model and the descriptors, which were evaluated on all three videos of the LITIV dataset. Globally, we see that our model ranks in second place across all methods, being surpassed only by mutual information (40×130). We believe this happens for a number of reasons. First of all, we reported the best results for each method, which were obtained with window sizes of 40 x 130. This window size is similar to a the size of a human silhouette in an image. Thus, these methods had more relevant information to work with than our method with patches of size $p_s \times p_s$ where $p_s = 37$. We believe this can lead to better recall results for mutual information [27]. In fact, with patches containing the same number of pixels as ours, about 1300, the results of mutual information are lower than ours with a recall of 0.649 for 10×130 patches. With twice the number of pixels as ours, that is 20×130 patches, our method is marginally better with a recall of 0.779 when compared to 0.775 of mutual information.

Second, if we look at table 1, we see that there are variations in the results of our method depending on the folds. Between the best fold and the worst one, there is a difference of almost 11%, with the other fold being in the middle of the two with a difference of close to 6% for the best fold and 5% with the worst one. If we refer to table 1 with all the folds data separation, we can see that the training set sizes vary between all folds. Although the difficulty of the test samples in the folds may vary, we remark that the best recall results were obtained on the fold with the most training samples available while worst results were obtained with less training data. This leads us to believe that with more data available, we would be able to get better results because, with the current folds, there is a correlation with the recall results and the amount of training data. This is why, in the case of the first fold, we are able to get slightly better result than mutual information, but that is not the case for

the second and third folds.

Even with achieving second place across all methods, we believe that our model is competitive with the handcrafted feature descriptors and with more data available, we could get event better performance. We believe this consists evidence that CNNs are able to, in the context of disparity estimation between visible and infrared domains, be competitive with the current feature descriptors.

5. Conclusions

In this paper, we presented a new model capable of doing disparity estimation between pairs of visible-infrared images. Our model is made of two subnetworks, each being a siamese network and outputting a log-probability vector for each possible disparity location. These two log-probability vectors are then summed together to get the final prediction. The goal of having two subnetworks working in parallel is to enforce left-right consistency in order to be more confident the final disparity prediction. We used two datasets, one used exclusively for training while evaluating on the other. Because of a lack of data, we did two data augmentation operations and performed a three-fold cross-validation. We demonstrated that we can achieve competitive performance with our proposed model when compared to handcrafted feature descriptors.

Acknowledgements

This work was supported by a scholarship from the National Sciences and Engineering Research Council of Canada (NSERC) and a grant from the Fonds de recherche du Quebec - Nature and Technologies (FRQNT). We thank NVIDIA Corporation for their donation of a Titan X GPU used for this research. Finally, a special thank you to Pierre-Luc St-Charles for his help rectifying the stereo images of his dataset.

References

[1] A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517, June 2012.

[2] G. Bilodeau, A. Torabi, and F. Morin. Visible and infrared image registration using trajectories and composite foreground images. *Image and Vision Computing*, 29(1):41 – 50, 2011.

[3] G.-A. Bilodeau, A. Torabi, P.-L. St-Charles, and D. Riahi. Thermal-visible registration of human silhouettes: A similarity measure performance evaluation. *Infrared Physics & Technology*, 64(C):79–86, 2014.

[4] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision – ECCV 2010*, pages 778–792, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[5] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018.

[6] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang. A deep visual correspondence embedding model for stereo matching costs. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 972–980, Dec 2015.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, June 2005.

[8] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, July 2011.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[10] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org, 2015.

[11] Z. Jie, P. Wang, Y. Ling, B. Zhao, Y. Wei, J. Feng, and W. Liu. Left-right comparative recurrent model for stereo matching. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[12] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[14] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004.

[15] W. Luo, A. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[16] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, USA, 2010. Omnipress.

[17] J. Pang, W. Sun, J. S. J. Ren, C. Yang, and Q. Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. *CoRR*, abs/1708.09204, 2017.

[18] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016.

[19] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[20] A. Shaked and L. Wolf. Improved stereo matching with constant highway networks and reflective loss. *arXiv preprint arxiv:1701.00165*, 2016.

[21] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[23] P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin. Online mutual foreground segmentation for multispectral stereo videos. *International Journal of Computer Vision*, Jan 2019.

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, June 2015.

[25] A. Torabi and G.-A. Bilodeau. Local self-similarity-based registration of human rois in pairs of stereo thermal-visible videos. *Pattern Recognition*, 46(2):578 – 589, 2013.

[26] A. Torabi, M. Najafianrazavi, and G. Bilodeau. A comparative evaluation of multimodal dense stereo correspondence measures. In *2011 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, pages 143–148, Sept 2011.

[27] P. Viola and W. M. Wells. Alignment by maximization of mutual information. In *Proceedings of IEEE International Conference on Computer Vision*, pages 16–23, June 1995.

[28] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4705–4713, Dec 2015.

[29] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Y. Choi. Action-decision networks for visual tracking with deep reinforcement learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1349–1358, July 2017.

[30] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17:1–32, 2016.