Object Tracking of Aerial Imaging Device Image Using Variational Autoencoder and External Memory

Keunho Park
Korea Electronics Technology Institute
Jeonju-si, Republic of Korea
root@keti.re.kr

Byoungjun Kim
Korea Electronics Technology Institute
Jeonju-si, Republic of Korea
jun0420@keti.re.kr

Donghoon Kim
Korea Electronics Technology Institute
Jeonju-si, Republic of Korea
clickmiss123@keti.re.kr

Seon-Hyeong Kim
Korea Electronics Technology Institute
Jeonju-si, Republic of Korea
sh.kim@keti.re.kr

Seo-jeong Kim
Korea Electronics Technology Institute
Jeonju-si, Republic of Korea
scott3554@keti.re.kr

Sunghwan Jeong
Korea Electronics Technology Institute
Jeonju-si, Republic of Korea
shjeong@keti.re.kr

***Abstract*— Object tracking is a fundamental problem in the field of computer vision. The object tracking methods proposed so far can be divided into a 'discriminative correlation filter' and a 'deep learning' based methods with a complex structure and a lot of computation. In this paper, we propose an algorithm for tracking objects with a simple structure while maintaining tracking performance using a convolutional variational auto-encoder, external memory, and a Siamese network. As a result of an experiment with an RT (real-time) data set to measure real-time, the result was a precision of 0.546 and a success rate of 0.527.**

Keywords— deep learning; object tracking; variational auto-encoder; external memory; Siamese network

I. INTRODUCTION

Object tracking, which automatically tracks a specified target in a changing video sequence, is a fundamental problem in many computer vision fields, such as visual surveillance, traffic monitoring, and automatic driving. The object tracking methods proposed so far can be largely divided into a 'discriminative correlation filter' and a 'deep neural network' based methods.

Existing object tracking algorithms have developed into complex structures to utilize image feature information or time-dependent information. In this paper, we propose a new object tracking method that surpasses the latest object tracking algorithms in a real-time object tracking environment by increasing the processing speed by introducing a simple and efficient algorithm as shown in Fig. 1. The proposed method can be applied to low computing power environments such as embedded environments when applied to aerial imaging devices such as vehicle image collecting device on the road or drones.

II. BACKGROUND THEORY

A. Variaational Auto-encoder

Variational auto-encoder (VAE) [1] inherited the characteristics of auto-encoder. However, the hidden layer h of the auto-encoder is simply a value that appears in the middle of the calculation without any special relation to the training data x, whereas the VAE has the difference that the hidden layer h is a random variable with a continuous distribution. The distribution of this hidden layer h is learned from data in the learning process. The hidden layer h has a probability distribution determined by the mean and standard deviation. As shown in Fig. 2, the VAE can be expressed as the probability $p(h|x)$ that the encoder obtains the hidden layer h from the given data x, and the probability $p(x|h)$ that the decoder obtains the data x from the hidden layer h.

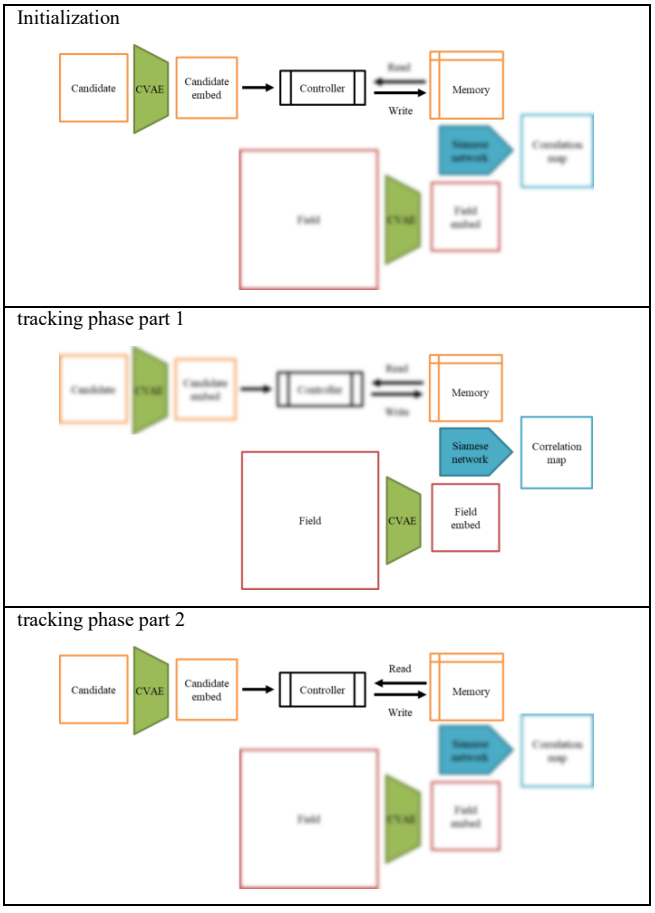


Fig. 1. Implementation of the proposed object tracking.

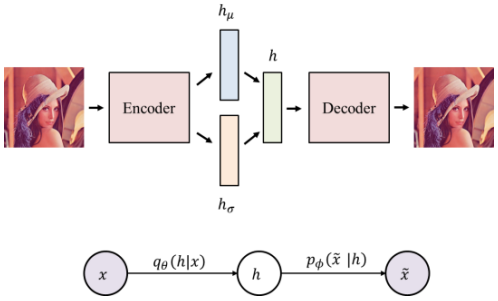


Fig. 2. Structure of a variable auto-encoder.

The decoder of the VAE is an artificial neural network that generates data x from the hidden layer h. To know the probability distribution $p(x)$ of the result x, the decoder learns $p(x|h)$. In terms of a manifold, $p(x|h)$ can be described as a function $f(x)=p(x|h)$ that projects an input x into the manifold space. Since the decoder is a generative model, it generates semantically similar results as the value of the hidden layer h changes.

The encoder of the VAE is an artificial neural network that obtains the hidden layer h from the input data x. Given the input data x, the probability distribution $p(h|x)$ of the hidden layer h is called the posterior distribution, and its value is difficult to calculate directly. Therefore, a method of approximating by $q(h|x)$ is used by introducing a variational inference method.

The goal of learning of VAE is to find the probability distribution of real data, $p(x)$. For convenience of calculation, if $p(x)$ is converted into a log format, the lower bound cost function of $\log p(x)$ using maximum likelihood estimation (MLE) $E(x)$:

$$\log p(x) \geq L(x)$$

$$E(x)=\mathrm{E}_{q_\phi(h|x)}\left[\log p(x|h)\right]-D_{KL}\left(q(h|x)\,\|\,p(h)\right)$$

The first term in Eq. (2) tells how efficiently the decoder reconstructs data x from the hidden layer h as a reconstruction error. The second term in Eq. (2) is the Kullback-Leibler divergence (K-L divergence) regularization term, which is a standard function that measures how different two different probability distributions $q(h|x)$ and $p(h)$ are. In variational inference, $q(h)$ is defined as a normal distribution.

$$q(h)=N\left(\mu_q,\sigma_q^2\right)$$

However, when the input data x is of high dimensionality, if q is defined as in Eq. (3), one normal distribution with the same mean and variance is assumed for all data, so it is very difficult to learn the network. To solve this problem, the VAE puts the parameters μ_q and σ_q of q as functions of x as follows.

$$q(h|x)=N\left(\mu_q(x),\Sigma_q(x)\right)$$

If the learning of the VAE is a simple maximum likelihood estimation problem, it can be solved with the artificial neural network's gradient-ascent algorithm. However, in order to put h as the input of the decoder, sampling is required because h is a random variable. To solve this problem, a method called re-parameterization is used. This method is a method of changing the stochastic property of the hidden layer h to deterministic by adding a random noise ε from the outside. Fig. 3 shows the re-parameterization process of the VAE. Diamond means a variable with a re-deterministic property, and a circle means a variable with a probabilistic property.

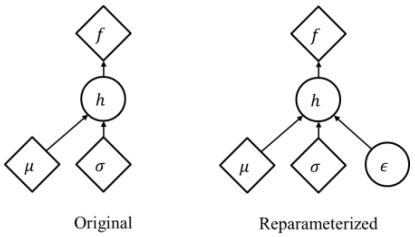


Fig. 3. Reparameterization of variable autoencoders.

B. *Neural Turing Machine*

A neural Turing machine (NTM) [2] refers to an artificial neural network that can be connected to an external memory. The structure of the NTM includes two components, a neural network controller and a memory bank, as shown in Fig. 4. Like most artificial neural networks, the controller interacts with the outside world through input and output vectors. Unlike standard artificial neural networks, it uses selective read and write operations to interact with the memory matrix. Similar to a Turing machine, the output of an artificial neural network can be variable using a 'head'. All components of a NTM can be differentiated, so we can train directly with gradient descent. Since the degree of interaction with the memory is sparse, the degree of interaction between read and write is determined by the weight size. Due to the read and write heads each have weights, each location can read and write as much information to memory as desired.

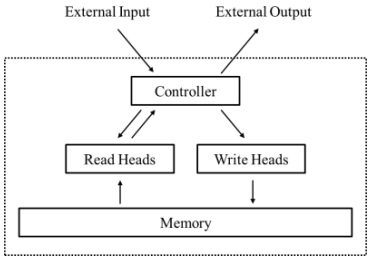


Fig. 4. Structure of a neural turing machine.

C. Siamese network

The Siamese network [3] has a structure as shown in Fig. 5, and consists of two identical sub-networks performing embedding and one cost function. The input to the network is a pair of samples and labels. Each input is embedded through a sub-network to produce two outputs. The cost function combines the label with the embedded output. The slope of the cost function for the parameter vectors controlling the two sub-networks is calculated through back propagation. The parameter vector is updated in a stochastic gradient descent method using the sum of the gradients provided by the two sub-networks.

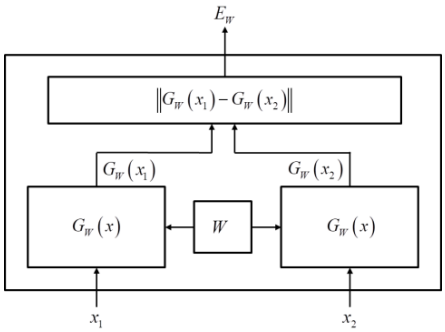


Fig. 5. The structure of the siamese network.

Let x_1 and x_2 be a pair of samples. If the samples x_1 and x_2 belong to the same class, y can be defined as $y=0$ as the binary label of the pair of samples; otherwise, it can be defined as $y=1$. Let W be the weight to be learned, and let the two points in the low-dimensional space created by projecting x_1 and x_2 be $G_W(x_1)$ and $G_W(x_2)$, respectively. Then, the cost function $E_W(x_1,x_2)$ that measures the similarity between the pair samples x_1 and x_2 can be defined as in Eq. (9).

$$E_W(x_1,x_2)=\|G_W(x_1)-G_W(x_2)\| \tag{9}$$

The Siamese network is a network that learns the embedding function G_W using a sample set $x=\{x_1,x_2,...,x_n\}$ by a deep learning method having multi-layers, and has a non-linear structure. Therefore, the analysis performance is superior to that of linear distance metric learning such as linear discriminant analysis.

III. Real-time Object Tracking Using CVAE and External Memory

In this chapter, we will describe a real-time object tracking method using a convolutional variational auto-encoder (CVAE), an external memory, and a Siamese network.

The structure and execution process of the proposed object tracking algorithm are shown in Fig. 1 and Table 1. The proposed object tracking algorithm takes as inputs a video sequence V, ground-truth gt, and the number of video sequences n. The proposed object tracking algorithm is divided into three steps: initialization (Lines 5 to 8 of Table 1), tracking phase part 1 (Lines 11 to 14 of Table 1), and tracking phase part 2 (Lines 15 to 17 of Table 1).

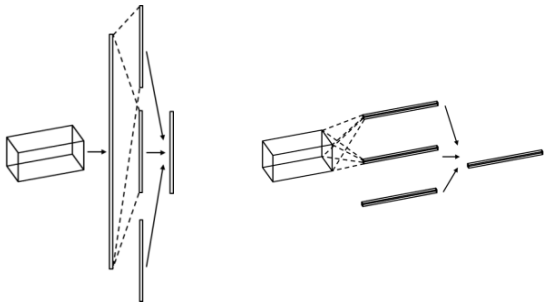
TABLE I. Pseudocode of The Proposed Object Tracking Algorithm

Notation	
V = video sequence	C = candidate image
n = # of video sequence	C_{embed} = embedded candidate image
gt = ground truth	M = external memory
b = result bounding box	S_F = size of field image
F = field image	S_C = size of candidate image
F_{embed} = embedded field image	$corr$ = correlation map
1	*procedure* **Proposed tracker**
2	**Input** : V , gt , n
3	**Output** : b
4	/* **Initialization** */
5	$b_1 = gt_1$
6	crop C from V_1 using b_1 with S_C
7	C_{embed} = VAE_embedding(C)
8	$M = C_{embed}$
9	/* **Tracking video frame** */
10	**for** $t = 2$ to n
11	crop F from V_t using b_{t-1} with S_F
12	F_{embed} = VAE_embedding(F)
13	$corr$ = siamese_network (F_{embed} , M)
14	$b_t = \arg\max_{x,y} corr$
15	crop C from V_t using b_t with S_C
16	C_{embed} = VAE_embedding(C)
17	M = Controller(M , C_{embed})
18	**return** b

A. Low-dimensional Embedding of Images Using CVAE

The CVAE proposed in this paper reduces the number of parameters to be trained by replacing the fully connected network (FCN) in the encoder and decoder of the VAE with a convolutional neural network (CNN). In addition, it has the advantage of being able to more effectively project the image onto the manifold space by extracting two-dimensional features using a convolution filter. In order to extract features from candidate and field images of different sizes using the VAE, all fully connected networks in the encoder part of the existing VAEs should be replaced with convolutional neural networks. Fig. 6(a) shows the sampling method used in the existing VAE. Transform the tensor of the last feature extraction layer of the VAE into a vector using flatten operation. After that, it is reduced to a vector of a desired dimension (d-dim) using a fully concatenated operation. In this way, the mean and standard deviation to obtain the probability $p(x)$ of the data are obtained, and the data are sampled using these. However, Fig. 6(b) is a method that allows sampling regardless of the size of the input image because it consists only of the convolution operation without the flatten operation. This makes it possible to apply the VAE to tracking objects based on Siamese networks with different candidates and field sizes. If

the convolution operation is performed on the tensor of the last feature extraction layer of the VAE using d filters having the same size as the tensor, a tensor with a size of $1\times1\times d$ can be obtained as a result. As in Fig. 6(a), the average and standard deviation of the probability of the data can be obtained and the data can be sampled using these.



(a) Sampling method of CVAE (b) CVAE's sampling method

Fig. 6. Two different sampling methods for VAE.

In order to explain the concept of the CVAE above, the case where the result of the convolutional encoder auto-encoder is a tensor of size $1\times1\times d$ was described. However, for images with high resolution, the dimensionality of the manifold space must also be increased. Therefore, it is possible to apply to the encoder structure of the VAE by changing all various convolutional neural networks such as ALexNet[4], LeNet[5], ZFNet[6], GoogleNet[7], VGGNet[8], ResNet[9], and DenseNet[10], which has proven the performance of feature extraction in previous studies, into a structure that can perform variational inference.

Due to the characteristics of video, even the same object exhibits various shapes, sizes, and colors over time due to factors such as lighting change, size change, occlusion, deformation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, and background confusion. However, the same temporally adjacent object has a very similar appearance despite the interference of various factors over time. Therefore, if the moving picture is divided into frames and projected onto the manifold space using the VAE, each frame will be located in a very close manifold space.

As shown in Fig. 7, the proposed CVAE structure uses the ALexNet variational inference method as an encoder and has a newly defined decoder attached. The CVAE is used for pre-training the parameters of the object tracking algorithm using the Siamese network.

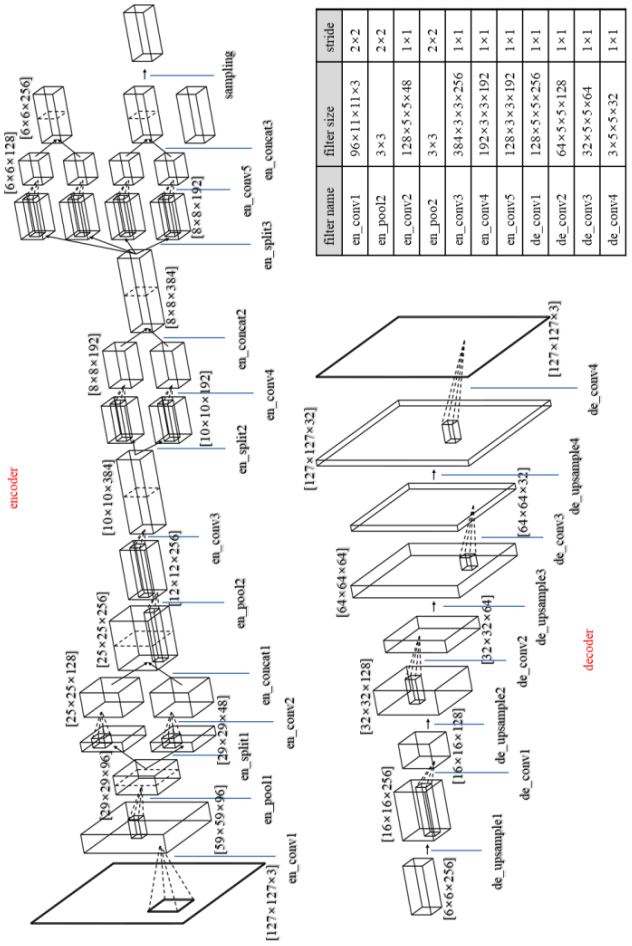
B. *External Memory Update*

Recurrent neural network (RNN), a kind of artificial neural network, has the concept of a kind of internal memory that remembers information according to time by introducing a directed circle that receives the output of the unit itself as input again. Therefore, RNN are introduced to object tracking to take advantage of these internal memory characteristics. However, since the RNN does not allow differential reception of information, it is difficult to track objects with properties such as long-term occlusion or out-of-screen. In this paper, we propose object tracking using an external memory that allows differential acceptance of information.

The proposed object tracking algorithm copies the low-dimensional embeddings of the candidate to the external memory in the initialization phase. However, when tracking starts, the controller updates the external memory with a constant weight ratio of the candidate's low-dimensional embeddings, as shown in line 17 of Table 1.

$$M_t[i] \leftarrow (1-w)M_{t-1}[i] + wC_t[i]$$

Where, $M_t[i]$ is the external memory value of position i at time t, $C_t[i]$ is the low-dimensional embedding value of the candidate at position i at time t, and w is the update weight.



filter name	filter size	stride
en_conv1	96×11×11×3	2×2
en_pool2	3×3	2×2
en_conv2	128×5×5×48	1×1
en_poo2	3×3	2×2
en_conv3	384×3×3×256	1×1
en_conv4	192×3×3×192	1×1
en_conv5	128×3×3×192	1×1
de_conv1	128×5×5×256	1×1
de_conv2	64×5×5×128	1×1
de_conv3	32×5×5×64	1×1
de_conv4	3×5×5×32	1×1

Fig. 7. Detailed structure of CVAE.

C. *Object Tracking Using Siamese Network*

The detected object of the moving picture $t-1$ frame is called a candidate, and the image of the t frame extracted within a sufficient range centered on the object position of the moving picture $t-1$ frame is called a field. Although the t-frame object is included in the field, it is difficult to include an

object of the same type as the environment changes with time as described above. Therefore, it is necessary to find the most similar part within the field as a candidate for the t-frame. As mentioned above, since the CVAE has a structure that removes all the structures of a fully connected neural network, it is possible to obtain low-dimensional embeddings of images of various sizes. When the candidate and the field pass through the CVAE, low-dimensional embeddings with the same number of dimensions but different widths and heights are extracted. Of these two features, the low-dimensional embedding of the volunteer is used to update the external memory. A correlation map can be obtained by inputting the updated external memory and performing a convolution operation using the low-dimensional embedding of the field as a filter. The part where the value of the correlation map is large is the part where the probability of the location of the object is high. This position becomes the position of the object of frame t.

IV. Experiments and Results

The experiment is performed on a RT (real time) dataset prepared by hand for real-time performance evaluation. For a fair comparison of the tracking results, an object tracking algorithm capable of direct experimentation was selected from among the object tracking algorithms with excellent performance and the experiment was conducted.

A. Dataset

The RT data set is a data set directly constructed to evaluate the real-time performance of the object tracking algorithm. As shown in Table 2, it is a data set obtained by extracting the images of objects with elements suitable for tracking among the videos collected at a height of 5m from the Honam Jeilmun Overpass located in Jeonju, Jeollabuk-do at the rates of 1FPS, 6FPS, and 30FPS.

TABLE II. Real-time Data Sets

Data Name	Video Length	Extraction FPS
CowShed1	1m 34s	1, 6, 30
CowShed2	1m	1, 6, 30
CowShed3	33s	1, 6, 30
HonamGate1	34s	1, 6, 30
HonamGate2	26s	1, 6, 30
HonamGate3	27s	1, 6, 30

B. Quantitative Evaluation Indices

Evaluation is based on two metrics: precision and success rate.

$$\text{precision}=\frac{\text{count}\left(\text{ED}\left(\text{gt\_bbox},\text{tr\_bbox}\right)<\text{threshold}\right)}{\text{count(total)}}$$

$$\text{success rate}=\frac{\text{count}\left(\text{IoU}\left(\text{gt\_bbox},\text{tr\_bbox}\right)>\text{threshold}\right)}{\text{count(total)}}$$

In Eq. 11 and 12, gt_box is the bounding box of the ground-truth, tr_box is the bounding box of the detection result, IoU is the intersection over union, and ED is Euclidean distance. As shown in Eq. 11, the precision shows the ratio of frames in which the distance between the center position of the boundary box of the actual measurement data and the center position of the boundary box of the tracking result is less than the threshold value. Representative precision is considered when the threshold value is 50 pixels. As shown in Eq. 12, the success rate represents the ratio of the intersection among the unions of the area of the bounding box of the actual measurement data and the detection result. In order to measure the real-time performance of the object tracking algorithm, the speed as well as the accuracy of the algorithm is a necessary comparison factor.

C. Experimental Environment

The object tracking algorithm proposed in this paper is implemented using Python's Tensor-Flow-Slim library. In addition, all experiments were performed in a PC environment equipped with intel(R) Core(TM) i7-6700k CPU @ 4.00GHz, and NVIDIA GeForce GTX TITAN X GPU and CUDA 9.0 installed. When the implementation language of various object tracking algorithms used for comparison is Python, the experiment was conducted in the same environment as the proposed algorithm, and when the implementation language was Matlab, it was performed in the environment of Visual Studio 2015 for Matlab R2018a and Mex C++ compiler use.

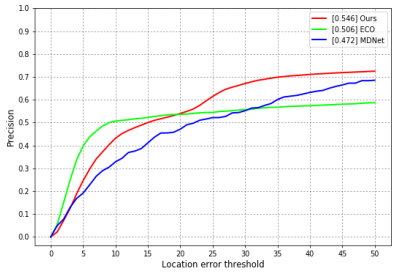
Fig. 8. HonamGate3 object tracking result image of RT data set.

D. *Experiment Result*

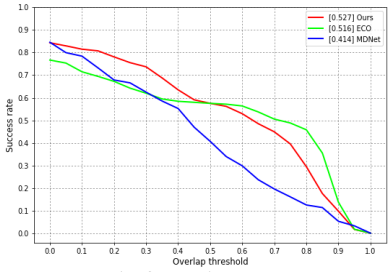
In the experimental process, MDNet[11], an object tracking algorithm with a speed of 1 FPS, is first tested with data with an image extraction FPS of 1 of the RT data set. Second, ECO[12] is tested with data with an image extraction FPS of 6 of the RT data set. Finally, the proposed object tracking algorithm, which has the best performance among the object tracking algorithms that crosses the real-time boundary, is tested with data with an image extraction FPS of 30 of the RT data set.

Fig. 8 is an object tracking result image of HonamGate3 of RT data set. In the case of HonamGate3, it can be seen that the proposed object tracking algorithm, ECO, and MDNet follow the passenger car properly until blockage occurs by the cargo truck in front of the passenger car, which is the object to be tracked. However, after the occlusion occurred, MDNet showed that the tracking object was lost. The reason why MDNet failed to track even though it was a relatively simple video is that the size change occurs as the object approaches the camera, but at 1FPS, the tracking speed of MDNet, the size change is believed to have adversely affected object tracking.

Fig. 9 shows the evaluation result of the object tracking algorithm using the RT data set. As a result of experimenting with data suitable for the speed of each object tracking algorithm, it can be confirmed that both MDNet and ECO object tracking algorithms have lower performance than the proposed object tracking algorithms.



(a) Precision plot of RT dataset



(b) Success rate plot for RT dataset

Fig. 9. Precision Plots and Success Rate Plots for the RT Dataset.

V. Conclusions and Future Challenges

In this paper, high-dimensional image data is converted into a low-dimensional embedding (manifold) using a CVAE to track the object of a video in real time. Algorithm for tracking an object by projecting it into space, storing this information using an external memory, and finding a correlation map, which is the degree of similarity between the external memory and object candidates, using a Siamese network has been proposed.

As a result of an experiment with an RT (real-time) dataset to measure real-time, the latest object tracking algorithm, ECO's precision 0.506, success rate 0.516, and MDNet's precision 0.472, success rate 0.414 and higher precision 0.546 and success rate 0.527 were obtained. It was proved that the real-time performance was excellent.

The object tracking algorithm proposed in this paper is a deep learning network trained using general-purpose video image data. The CVAE, which is a component of the network, has the ability to project data with a limited domain, such as a human face, a car, or a designated livestock, into a manifold. Therefore, in order to apply the proposed object tracking algorithm to a specific field, it is expected that it can be used as an object tracking network specialized in that field if it is re-learned and applied using fine-tuning learning with a data set corresponding to that field.

Acknowledgment

This work was supported by Korea Institute of Police Technology(KIPoT) grant funded by the Korea government(KNPA) (No.092021C28S02000, Development of traffic information shadow section information generation and operation management technology for introduction of cooperative traffic control strategy)

References

[1] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” in Proceedings of the International Conference on Learning Representations, 2014.

[2] A. Graves, G. Wayne and I. Danihelka, “Neural turing ma-chines,” in *CoRR*, 2014.

[3] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi and P. H. S. Torr, “Fully-convolutional Siamese networks for object,” in *ECCV workshop*, 2016.

[4] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," 2012.

[5] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," vol. 86, no. 11, pp. 2278-2324, 1998.

[6] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," in *ECCV 2014*, 2014.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Go-ing deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, 2015.

[8] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR,* 2015.

[9] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learn-ing for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition,* pp. 770-778, 2016.

[10] G. Huang, Z. Liu, L. V. D. Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *2017 IEEE Conference on Computer Vision and Pattern Recogni-tion*, Honolulu, HI, 2017.

[11] H. Nam and B. Han, “Learning Multi-Domain Convolutional Neural Networks for Visual Tracking," in *2016 IEEE Conference on Computer Vision and Pattern Recognition*, June, 2016.

[12] M. Danelljan, G. Bhat, F. Shahbaz Khan and M. Felsberg, "Efficient Convolution Operators for Tracking," in *2017 IEEE Conference on Computer Vision and Pattern Recogni-tion*, Honolulu, HI, 2017.