Instance-Level Future Motion Estimation in a Single Image Based on Ordinal Regression

Kyung-Rae Kim[1] Whan Choi[1] Yeong Jun Koh[2] Seong-Gyun Jeong[3] Chang-Su Kim[1]
[1]Korea University [2]Chungnam National University [3]CODE42.ai

Abstract

A novel algorithm to estimate instance-level future motion in a single image is proposed in this paper. We first represent the future motion of an instance with its direction, speed, and action classes. Then, we develop a deep neural network that exploits different levels of semantic information to perform the future motion estimation. For effective future motion classification, we adopt ordinal regression. Especially, we develop the cyclic ordinal regression scheme using binary classifiers. Experiments demonstrate that the proposed algorithm provides reliable performance and thus can be used effectively for vision applications, including single and multi object tracking. Furthermore, we release the future motion (FM) dataset, collected from diverse sources and annotated manually, as a benchmark for single-image future motion estimation.

1. Introduction

Motion understanding is critical in various vision tasks, such as optical flow [23, 38, 46], action recognition [55], future frame prediction [36], and video compression [50]. Most prior arts estimate motions by analyzing the differences between consecutive frames. In contrast, a human being is often capable of anticipating motions precisely even from a single still image, as illustrated in Figure 1. Such innate perceptual capabilities enable us to take desired actions and avoid dangerous situations. If computer vision attains a similar level of motion understanding, we would be able to build more secure artificial intelligent systems, *e.g.* robots and self-driving cars.

We propose a pioneering algorithm to estimate instance-level FM in a single image. The proposed algorithm attempts to challenge humans in single-image motion understanding. Even though there are some methods [16, 26, 37, 40, 52] for predicting FM, they need additional information or are effective in limited environments only. As shown in Figure 2, Yagi *et al*. [52] require a cumulative trajectory of an instance from past frames, while the proposed algorithm uses only a present frame. Also, [26, 37] estimate object trajectories in a scene. Given starting and end points in a scene, they estimate trajectories connecting those points. Mottaghi *et al*. [40] estimate motion scenarios by classifying objects in images into one of pre-defined scenarios as in Figure 2. They focus on the scene understanding task rather than on object FMs, such as motion directions and magnitudes. Gao *et al*.'s algorithm [16] is the most similar to ours, but it estimates pixel-level optical flow and works only for highly similar action scenes to the ones in the training data.

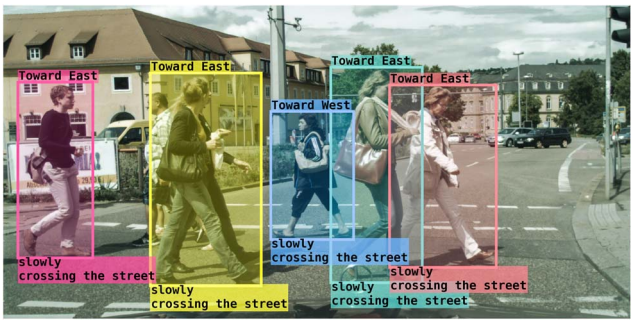


Figure 1. Above annotations are automatically generated from the single image by the proposed algorithm.

Even unaware of the exact physics, humans can predict next motions of instances based on their experience. Based on this observation, we use a deep neural network [22] to implement such perceptual capabilities regarding FM. Deep neural networks can perform high-level understanding of images, if reliable examples are provided sufficiently [17, 20, 24, 29–31, 53]. Hence, another objective of this work is to construct a reliable dataset for single-image FM estimation. First, we focus on pedestrian instances, which are objects of the most interest in many applications. We construct the FM dataset by collecting images, containing pedestrians. After detecting bounding boxes for pedestrians, we manually assign three attributes of FM (*i.e.* direction, speed, and action) to each pedestrian. Later, we extend the FM dataset to include car and animal instances.

When we see a moving object, we perceive visual information of the object and its surroundings simultaneously. Similarly, to exploit scene contexts for FM estimation, we propose the multi-context pooling (MCP) layer that inte-

Yagi *et al*. [52] Kitani *et al*. [26] Ma *et al*. [37] Mottaghi *et al*. [40] Gao *et al*. [16] Proposed

Figure 2. Comparison of previous FM estimation methods [16, 26, 37, 40, 52] and the proposed algorithm: the proposed algorithm estimates FM in a single image more reliably in more diverse scenes and environments. Please also see the supplemental video.

grates both object and global features. We incorporate the MCP layer into DenseNet-121 [22] to learn a unified model for estimating the future direction, speed, and action of an instance. Moreover, for effective FM estimation, we adopt ordinal regression. Especially, we propose the cyclic ordinal regression (COR) scheme for the future direction, whose classes have a cyclic order.

The proposed algorithm provides promising FM estimation results despite variations in camera viewpoints, source types, and environments. We assess the efficacy of the proposed algorithm in important applications of single and multi object tracking. It is demonstrated that the proposed algorithm makes conventional object trackers [25, 41] more efficient by reducing search regions. Moreover, we extend the proposed algorithm to process other kinds of instances, including cars, cats, dogs, and horses.

This work has the following main contributions:

- We develop a single-image FM estimation algorithm, by incorporating the MCP layer into DenseNet-121 and developing the COR scheme for future direction classification.
- Unlike the previous attempts [16, 26, 37, 47, 52], the proposed algorithm estimates FM reliably in diverse scenes and environments. It is strongly recommended to see the supplemental video.
- We demonstrate the efficacy of the proposed algorithm quantitatively in single and multi object tracking.
- We release the FM dataset to serve as a benchmark for the interesting research topic of subsequent behaviour estimation in a single still image.

2. Related Work

2.1. Instance-Level Future Motion Estimation

FM estimation can be performed at either instance level or pixel level. Let us first review instance-level algorithms. Mottaghi *et al*. [40] forecast FMs of objects in a single image. They pre-define 66 scenarios based on physical movements of an object, and classify an object into one of the scenarios. Chao *et al*. [4] forecast human poses from static images. They design recurrent neural networks to yield a pose sequence from an initial pose. However, it works for limited sports scenes only.

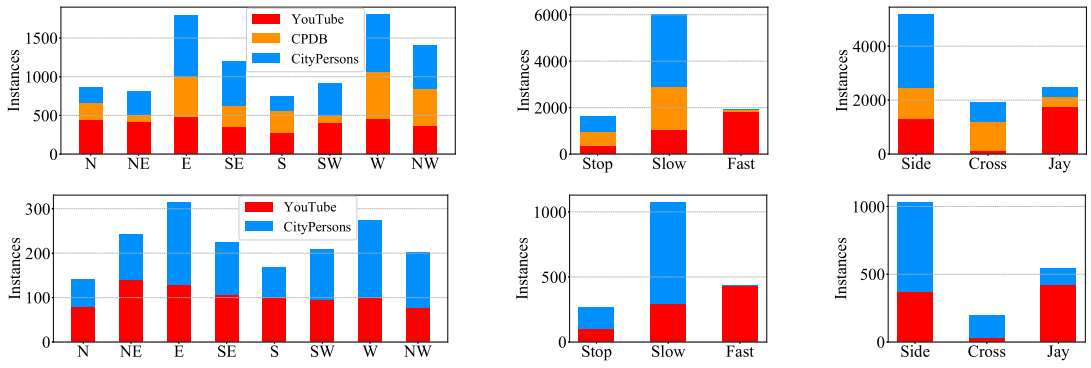
Also, some methods predict long-term trajectories of objects in a still image [26, 37, 47]. Using a Markov decision process, Kitani *et al*. [26] predict trajectories of an object based on its current and goal states. Ma *et al*. [37] extract visual attributes of pedestrians via deep networks to forecast pedestrian dynamics. Walker *et al*. [47] exploit mid-level patches for future trajectory prediction. However, since they focus on scene information rather than on object information, these conventional techniques are highly dependent on scene types and camera angles. In contrast, we attempt to design a domain-adaptive FM estimation algorithm, which can be reliably used in various applications.

2.2. Pixel-Level Future Motion Estimation

Some algorithms [16, 34, 43, 48, 49] estimate dense motion, *i.e*. optical flow, from a single image. Pintea *et al*. [43] predict continuous motion vectors using structured random forests with regression. Walker *et al*. [48] quantize optical flow vectors into 40 clusters and then classify each pixel into one of the clusters using convolutional neural networks. Using a conditional variational autoencoder (VAE), Walker *et al*. [49] characterize various distributions of future flow to predict multiple motion directions and magnitudes at each pixel. Gao *et al*. [16] develop an encoder-decoder network with a loss network to hallucinate future flow and demonstrate the efficacy of the hallucinated flow for action recognition. Li *et al*. [34] devise a spatio-temporal conditional VAE to predict future flow maps in multiple time steps. With the predicted maps, they perform full frame synthesis and achieve video prediction.

2.3. Ordinal Regression

Ordinal regression is a learning task for predicting a label (or rank) of an object, where the set of labels has a linear order [21], *e.g*. the set of integers. Attempts have been made for ordinal regression [19], for example, using support vector machines [6], Gaussian processes [5], and perceptron learning [8]. Frank and Hall proposed the ordinal binary decomposition [15]. They constructed independent binary classifiers to decide whether the rank of an object is greater than k. Using decision trees, they combined the binary outputs to estimate the rank. Li and Lin [33] proposed a reduction scheme from ordinal regression to binary classification. A set of binary classifiers was learned jointly and combined with existing techniques, such as SVM. This



(a) Direction (b) Speed (c) Action

Figure 3. Statistics of the FM dataset. The instances are sampled from YouTube [1], CityPersons [54], and CPDB [13], and then split into training (top) and test (bottom) sets.

reduction scheme has been adopted in various applications, including the age estimators in [3,42]. Also, the decomposition for the case of a cyclic order was considered in [11]. In this work, we formulate the FM classification as an ordinal regression problem.

3. Single-Image Future Motion Estimation

We represent the FM of an instance with its direction, speed, and action classes. We then develop classifiers, based on ordinal regression, for the FM estimation.

3.1. FM Dataset

Motion information is often represented by displacement vectors between successive frames. For example, dense optical flow estimates pixel-level correspondence between video frames. However, it is hard to extract pixel-level motion information precisely and reliably from videos. Instead, through relatively easy annotations, we collect instance-level motions in still images, which are sampled from existing datasets [7, 13, 54] and YouTube [1]. This instance-level annotation facilitates the construction of a large dataset. Moreover, the instance-level motion estimation can be performed reliably using a deep neural network.

The proposed algorithm is not limited to a particular type of instances, but we first focus on pedestrians for the following reasons. First, it is easy to access public datasets, capturing street scenes, and annotate lots of pedestrians. Second, excluding abnormal situations, human behaviour is predictable even in a single image. In other words, pedestrians' FMs can be inferred from semantic contexts in general. Third, humans are often objects of the most interest.

The FM dataset annotates 11,342 pedestrian instances. In YouTube videos, we have no ground-truth bounding boxes of instances. Thus, we run the YOLOv3 detector [44] to obtain the bounding boxes. For each instance, we label its direction, speed, and action classes manually by referring to the current frame and nine subsequent frames. We quantize

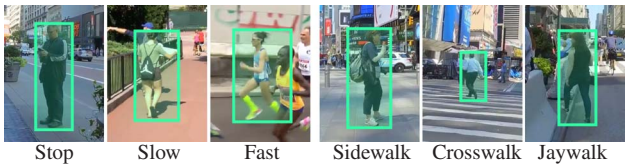


Figure 4. Three speed classes and three action classes.

the future direction into one of the four cardinal directions (N, E, S, W) and the four intermediate ones (NE, SE, SW, NW) in the image coordinates. We use the eight quantized directions, which are sufficient in many applications. Finer quantization makes the annotation difficult and unreliable. For a similar reason, we have three speed classes of 'stop,' 'slow,' and 'fast.' Action classes can be varied in applications. In this work, to monitor pedestrians' behaviour on streets, we define three action classes of 'sidewalk,' 'crosswalk,' and 'jaywalk,' as shown in Figure 4.

Figure 3 shows the class distributions of instances in the FM dataset. These instances are sampled from YouTube [1], the CityPersons dataset [54], and the Caltech Pedestrian Detection Benchmark (CPDB) dataset [13]. We split the entire dataset into training and test sets with the ratio of 0.85 to 0.15. We use the CPDB dataset for training only, since it has fewer objects than YouTube or CityPersons and more than 75% of the objects are contained in only 3 videos.

3.2. Future Motion Network

We develop a deep neural network for the FM estimation, which performs three classification tasks for direction, speed, and action. The network processes an image patch, in which a pedestrian is located at the center, and yields the three classification results. It is composed of a feature extractor and a classifier, as shown in Figure 5.

We assume that, in an image, pedestrians are either located manually or parsed by an object detector. Suppose that a pedestrian has a bounding box with height h. Then, around the bounding box, we crop the $2h \times 2h$ patch, which is put into the network. The feature extractor then yields

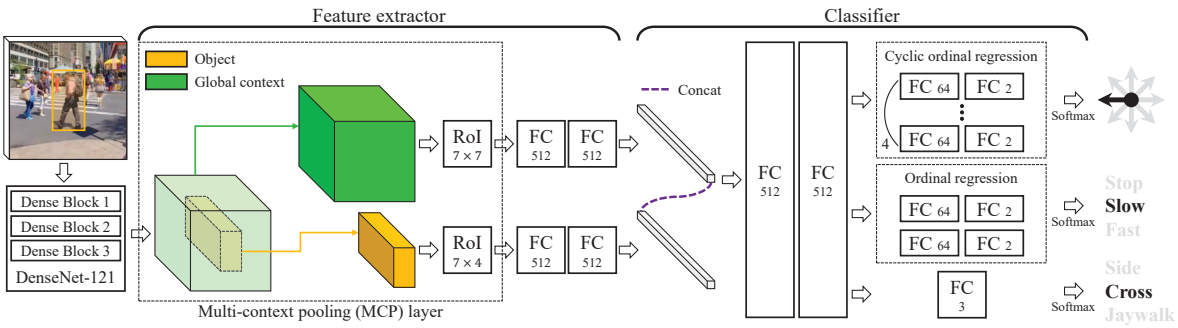


Figure 5. The architecture of the proposed FM network. Three dense blocks of DenseNet-121 [22] are used.

object and global features based on DenseNet-121 [22]. To extract those features from the output of DenseNet-121, we develop the MCP layer, which uses two region of interest (RoI) pooling layers:

- The bounding box is the RoI for an object feature, which is pooled to spatial resolution 7×4. The object feature conveys appearance information of a pedestrian with minimal background.
- The $2h \times 2h$ patch is the RoI for a global context feature, pooled to resolution 7×7. This global feature is also important in FM estimation, since it provides overall semantic information about a scene.

We determine the output sizes of the RoI pooling layers empirically. Each RoI pooling layer is followed by two fully-connected (FC) layers.

Then, the classifier performs the three classification tasks, by employing FC layers and softmax layers. Specifically, the object and global features are concatenated and processed by two FC layers and three sub-classifiers for the pedestrian's direction, speed, and action. The three sub-classifiers are designed differently. First, we classify the direction using the COR scheme in Section 3.3. Second, we do the speed using the linear ordinal regression [33], since the speed classes are in the order of 'stop,' 'slow,' and 'fast.' In other words, 'stop' and 'fast' are more different from each other than 'stop' and 'slow' are. Third, we perform the 3-way classification of the action using a softmax layer, since there is no ordinal relation among action classes.

3.3. Cyclic Ordinal Regression

As shown in Figure 6, the future direction of an object is classified into one of the eight directions: N (c_0), NE (c_1), E (c_2), SE (c_3), S (c_4), SW (c_5), W (c_6), NW (c_7). The direction classes have a cyclic order, since N (c_0) is adjacent to both NW (c_7) and NE (c_1). Note that many physical quantities have cyclic orders, *e.g.* 24 hours in a day, longitudes, as well as directions on a plane. Suppose that there are K directional classes in a cyclic order,

$$\mathcal{C} = \{c_0, c_1, \ldots, c_{K-1}\} \tag{1}$$

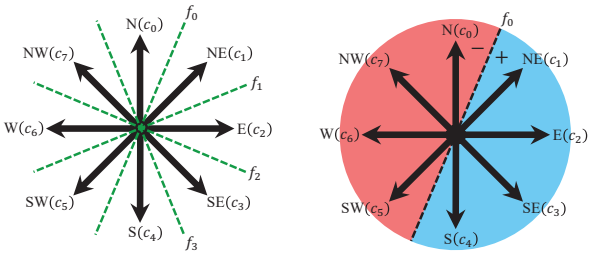


Figure 6. Binary classifiers for the cyclic ordinal regression.

where K is an even number. In such a case, it is not desirable to apply the K-way classification that does not consider the cyclic order in the loss function. For example, if direction N is misclassified into S, the error is severer than its misclassification into NE. Therefore, we propose the COR scheme, by extending the ordinal binary decomposition technique in [33].

Let x be an instance and $y_x \in \mathcal{C}$ be its class. For COR, we use binary classifiers, $f_0, f_1, \ldots, f_{K/2-1}$. Each binary classifier f_n is defined as

$$f_n(x) = \begin{cases} 1 & \text{if } y_x \in \{c_{(n+1)_K}, \ldots, c_{(n+K/2)_K}\} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $(n)_K$ denotes the remainder after the division of n by K. In other words, f_n divides $\mathcal{C}$ into two subsets of the same size, and determines whether the class of x is between $c_{(n+1)_K}$ and $c_{(n+K/2)_K}$ or not. In Figure 6, f_0 halves the eight directions into the blue and red sides. It outputs 1 if the direction is NE, E, SE, or S, and 0 otherwise.

From (2), it can be shown that

$$f_n = 1 - f_{n+K/2}, \tag{3}$$

$$f_n = f_{n+K}. \tag{4}$$

Due to the symmetry and periodicity, all classifiers are determined by only $K/2$ classifiers, $f_0, f_1, \ldots, f_{K/2-1}$. Note that, in the linear ordinal regression [33], the classes in a line segment is divided into two parts. Therefore, for K-way classification, $K-1$ binary classifiers are required. In contrast, in the proposed COR, a circle is halved into two semicircles, as done in [11]. Thus, only $K/2$ binary classifiers are needed.

During the training of the classifiers, f_n is assigned a binary ground truth value in (2). On the other hand, in testing, f_n yields a confidence value (*i.e.* softmax probability) between 0 and 1. Using these confidence values of the $K/2$ classifiers, class c_{k^*} of instance x is determined by

$$k^* = \underset{k \in \mathcal{C}}{\operatorname{argmax}} \sum_{n=1}^{K/2} f_{k-n}(x) \tag{5}$$

For example, suppose that $K = 8$ as in Figure 6. Ideally, when x has class c_2,

$$\begin{aligned} \sum_{n=1}^{4} f_{2-n}(x) &= f_1(x) + f_0(x) + f_{-1}(x) + f_{-2}(x) \\ &= f_1(x) + f_0(x) + 1 - f_3(x) + 1 - f_2(x) = 4 \end{aligned}$$

which is greater than or equal to $\sum_{n=1}^{4} f_{k-n}(x)$ for all k. Thus, its class is declared correctly as c_2. Also, it can be shown that (5) is the maximum likelihood (ML) decision rule [14], if each $f_{k-n}(x)$ represents the probability that x has one of the four directions as defined in (2).

3.4. Learning Network

The MCP layer connects DenseNet-121 and the FC layers using the two RoI pooling layers. Thus, the network in Figure 5 can accept a patch of an arbitrary size. However, for effective training and inference, we normalize the size of a patch to 400×400 so that it contains a pedestrian at the center whose height is 200 pixels.

We define the overall loss function as

$$L = L_{\text{Dir}} + L_{\text{Spe}} + L_{\text{Act}} \tag{6}$$

where L_{Dir}, L_{Spe}, and L_{Act} are the losses for the direction, speed, and action classification, respectively. For L_{Dir}, we adopt the sum of binary cross entropies [42]. Specifically,

$$L_{\text{Dir}}(\mathbf{p}, \mathbf{q}) = -\sum_{n=0}^{3} \sum_{i=0}^{1} q_n^i \log p_n^i \tag{7}$$

where $\mathbf{p} = \{p_n^i : i = 0, 1 \text{ and } n = 0, 1, 2, 3\}$ is the softmax probability vector from the four binary classifiers f_n and $\mathbf{q} = \{q_n^i\}$ is the corresponding ground-truth binary vector. L_{Spe} is defined similarly. L_{Act} is defined as

$$L_{\text{Act}}(\mathbf{p}, \mathbf{q}) = -\sum_{i=1}^{3} q_i \log p_i \tag{8}$$

where $\mathbf{p} = \{p_i\}$ is the softmax probability vector for the three actions and $\mathbf{q} = \{q_i\}$ is the ground-truth binary vector.

We train the network via the stochastic gradient descent with a momentum of 0.9 and a batch size of 4 for 20 epochs. The learning rate is 10^{-4} for the first ten epochs and 10^{-5} for the last ten epochs. As initial parameters, we use the DenseNet-121 model [22] pre-trained on ImageNet [10].

Table 1. Classification accuracies (%) according to variations in the MCP layer and the ordinal regression.

	Direction	Direction+	Speed	Action
Object	75.9	95.7	88.3	86.8
Global	25.8	45.7	82.1	85.5
Object + Global	76.8	96.0	90.1	86.7
Proposed	**77.7**	**96.6**	**90.4**	**87.2**

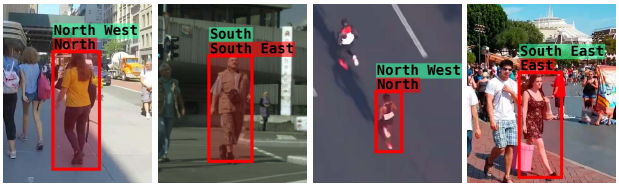


Figure 7. Direction classification: Green labels are the ground-truth, while red ones are predicted directions. In these cases, the ground-truth and predicted directions are adjacent to each other.

3.5. Experimental Results

Table 1 is an ablation study on the MCP layer and the ordinal regression scheme. 'Direction,' 'Speed,' and 'Action' are the classification accuracies of the FM direction, the FM speed, and the FM action, respectively. For the first three configurations, the multi-class classification is used instead of the ordinal regression. 'Object' and 'Global' denote the networks, in which only object and global context features are used for the classification, respectively. Note that 'Object + Global,' combining both features, outperforms both 'Object' and 'Global' with the exception of the action classification. The last configuration, 'Proposed' uses the linear ordinal regression for the speed classification and the COR scheme for the direction classification. It outperforms all the other configurations in all three classification tasks. Even in the action classification, it improves the performance, since the network is trained to extract more effective features. Notice that the direction classification is more challenging than the speed and action classification, in which the accuracies are about 90%.

In Table 1, 'Direction+' means the accuracy when the estimated direction is regarded as correct if it is identical with or adjacent to the ground-truth direction. For example, for the ground-truth direction N, an estimated direction NE, N, or NW is correct in the 'Direction+' accuracy. Figure 7 shows examples, in which the ground-truth and predicted directions are adjacent. In these examples, even a human being cannot easily quantize the true direction into one of the two classes by looking at the single image only. Thus, 'Direction+' takes into account this ambiguity. The proposed algorithm yields the 'Direction+' accuracy of 96.6%.

As mentioned previously, there is no existing algorithm, exactly matching the proposed algorithm. Thus, for comparison, we implement future direction classifiers using handcrafted features or CNN features. In Table 2, we use HOG [9] and ACF [12] as handcrafted features for pedes-

Correct estimation

False direction False speed False action

Figure 8. FM estimation results. The first row presents correct estimation results. The second row shows failure cases, where top labels are the ground-truth and bottom ones are predicted classes.

trians. We extract CNN-1 features by adopting VGG-16 in Faster R-CNN [45]. For CNN-2 features, we use DenseNet-121 [22], trained on ImageNet [10]. For these features, we classify the future direction using SVMs. For Gao *et al*. [16], we obtain pixel-level flow vectors and aggregate them to get instance-level motion vectors. The proposed algorithm outperforms these comparison methods significantly. Gao *et al*. [16] can predict pixel-level flows for specific actions scenes, as shown in Figure 2, but fail to provide reliable flows on the FM dataset. Thus, its instance-level direction estimates are almost random. Here, we do not retrain the network in [16] on the FM dataset. Gao *et al*. [16] requires reliable optical flow vectors for its training, but the FM dataset provides sparsely sampled frames only.

Figure 8 presents FM estimation results. In the top row, FMs are predicted successfully even when the scenes are crowded or cluttered. The bottom row shows failure cases: The direction is falsely predicted, since the pedestrian is far away and it is unclear whether he faces the camera or moves away in the opposite direction. In the false speed case, three people stand still in fact but are declared to walk slowly. The middle person is more challenging. We can learn that people looking at cell phones usually do not walk, but it is hard to tell whether the middle person stands still or walks. In the false action case, since the pedestrian is surrounded by many people, he is falsely claimed to be on a sidewalk.

4. Applications

For applications of the proposed FM estimation algorithm, it is strongly recommended to see the supplemental video and document. For example, it is demonstrated that the action classification of 'sidewalk,' 'crosswalk,' and 'jaywalk' can assist autonomous driving and improve the safety of pedestrians. Moreover, the proposed algorithm can be used for the crowd analysis in a single image.

Table 2. Classification accuracies (%) of the future direction.

	HOG [9]	ACF [12]	CNN-1 [45]	CNN-2 [22]	Gao *et al*. [16]	Proposed
Direction	43.1	42.0	47.1	31.9	12.4	**77.7**
Direction+	72.2	72.3	72.7	59.8	38.5	**96.6**

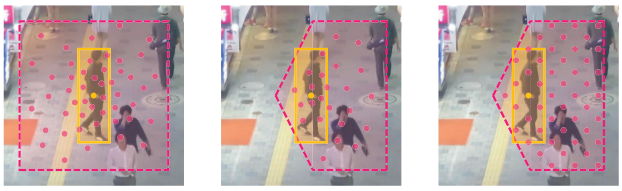
Among many possible applications, this section introduces two applications of the proposed algorithm: single and multi object tracking. By exploiting 'FM direction' and 'FM speed', we make the conventional object trackers [25, 41] more efficient.

4.1. Single Object Tracking

By reducing a search region for a target object using its FM, we can track the object more efficiently. We adopt MDNet [41] as the baseline single object tracker, which exhibits competitive performances in several tracking benchmarks [27, 28, 51]. The baseline selects search candidates within a square window, by sampling from a Gaussian distribution. The side length of the square depends on the size of an object. We follow the details in [41].

We use a predicted FM to narrow the search region. When the FM speed is 'stop,' we adopt the four times smaller square than the baseline. Otherwise, we determine a fan-shaped area in the direction of the FM. We set the angle of the fan-shaped area to 135°, but the maximum distance from the target to a search candidate is kept the same as that of the baseline. Figure 9 compares the sampling strategies of the baseline MDNet and the proposed 'MDNet + FM.'

We evaluate the performance of MDNet+FM on the temple color 128 (TC128) dataset [35] and the object tracking benchmark (OTB) dataset [51]. Since we focus on



(a) (b) (c)

Figure 9. Comparison of sampled search points, depicted by red dots, in (a) MDNet, (b) MDNet+FM, and (c) CDT+FM. The points in (a) and (b) are sampled from a Gaussian distribution, while those in (c) are from a uniform distribution.

Table 3. Comparison of tracking performances of MDNet [41] and MDNet+FM on pedestrian sequences in the TC128 and OTB datasets.

Setting	Method	# Samples	PR	SR	fps
I	MDNet	128	0.845	0.589	1.67
	MDNet+FM	85	**0.848**	**0.598**	**2.75**
II	MDNet	192	**0.871**	**0.614**	1.51
	MDNet+FM	128	0.849	0.595	**2.45**
III	MDNet	256	**0.883**	**0.616**	1.41
	MDNet+FM	171	0.830	0.582	**2.24**
IV	MDNet	320	0.875	0.618	1.23
	MDNet+FM	213	**0.893**	**0.623**	**2.15**
V	MDNet	384	0.837	0.584	1.13
	MDNet+FM	256	**0.872**	**0.613**	**1.98**

pedestrian instances, we use only the sequences for tracking pedestrians. TC128 and OTB have 23 and 22 such sequences, respectively. After removing duplicated ones, there are 33 pedestrian sequences in total. To measure the tracking performance quantitatively, we use precision (PR) and success rate (SR) [41].

Table 3 compares the performances of MDNet+FM and the baseline MDNet. '# Samples' is the number of search candidates. Both MDNet and MDNet+FM use the same Gaussian sampling, but MDNet+FM reduces the search region. Thus, in the same setting in Table 3, MDNet+FM searches substantially fewer candidates than MDNet, improving the tracking speed. However, MDNet+FM provides comparable or even better tracking performances than MDNet. For example, in setting IV, MDNet+FM provides slightly higher PR and SR scores than MDNet, as well as it is 175% faster. Figure 10 plots PR and SR scores versus fps. Again, when we compare the results with similar PR or SR scores, MDNet+FM is significantly faster than the baseline.

4.2. Multiple Object Tracking

Objects in multiple object tracking (MOT) sequences [39] tend to exhibit slow and smooth motions between successive frames. To exploit this property, Bochinski *et al*. [2] proposed an MOT algorithm, which makes a decision using only the intersection-over-union (IOU) ratio between the bounding boxes of a target object and a search candidate. However, in a low frame rate video, *e.g*. less than 10 frames per second (fps), their algorithm may fail since there

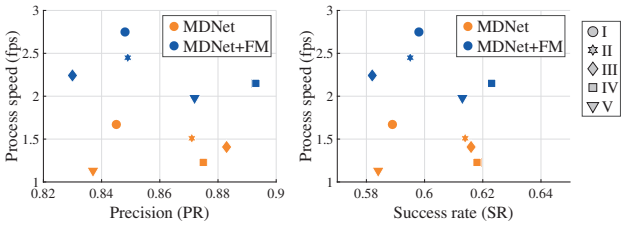
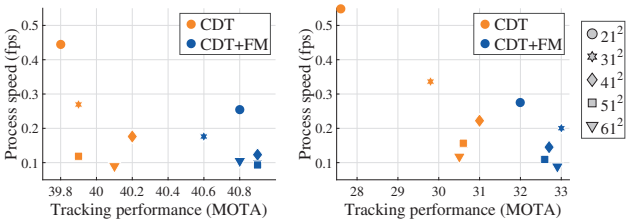


Figure 10. PR and SR scores versus tracking speeds on pedestrian sequences in the TC128 and OTB datasets.



(a) (b)

Figure 11. MOTA scores versus tracking speeds on the MOT17 dataset at video frame rates of (a) 5 fps and (b) 1 fps.

can be abrupt changes between frames. For successful MOT in low frame rate videos, we apply the proposed FM algorithm to a more sophisticated MOT algorithm, CDT [25], which is a tracking-by-detection method.

We reduce the search region of CDT using the FM of a target object, as shown in Figure 9(c). In CDT, a uniform distribution is used for sampling search points, and the number of search points is set to 21^2, 31^2, 41^2, 51^2, or 61^2. For CDT+FM, we reduce the search region but increase the sampling density to have the same number of search points. We use the MOT17 benchmark [39]. Four out of seven sequences in MOT17 have camera movements, which make the search range reduction using FM invalid. Therefore, we apply the background compensation to all sequences, by employing the BRISK keypoint matching [32] and an affine transformation. Then, we evaluate the MOT accuracy (MOTA), which is one of the most comprehensive metrics in the benchmark [39].

We present a table in the supplemental document to compare CDT+FM with the baseline CDT in detail. In Figure 11, we plot MOTA scores versus processing speeds according to the numbers of search points. At similar processing speeds, CDT+FM provides a significantly higher MOTA than CDT. At the same number of search points, CDT+FM yields a slower processing speed, since it performs the background motion compensation. However, by reducing the search region based on FM, it provides more accurate tracking results, yielding a higher MOTA score.

5. Other Kinds of Instances

The proposed single-image FM estimation algorithm can be applied to not only pedestrians but also other kinds of instances. This section extends the proposed algorithm to

Figure 12. FM estimation of cars. The left two images show correct results. The right two images are failure cases for the direction classification and the action classification, respectively, where green labels are the ground-truth and red labels are predicted classes.

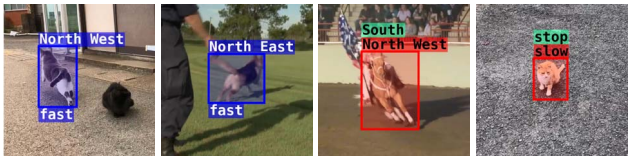


Figure 13. FM estimation of animals. The two left images show correct results. The two right images are failure cases, where green labels are the ground-truth and red ones are predicted classes.

Table 4. Classification accuracies (%) of the proposed single-image FM estimation on car and animal instances.

Instance	Method	Direction	Direction+	Speed	Action
Car	Baseline	89.6	98.5	97.0	94.2
	Proposed	**89.9**	**98.7**	**97.4**	**94.5**
Animal	Baseline	87.1	97.1	74.2	-
	Proposed	**87.6**	**97.4**	**74.9**	-

handle two more kinds: cars and animals. These instances have different characteristics from pedestrians. Thus, although we use the same network architecture in Figure 5, we train the parameters separately for cars and animals. Also, we define the classes in different ways.

For cars, there are 8 directional classes in the same way as pedestrians. In the case of speed, even a human being cannot easily predict the absolute speed of a car in a single image, since the car has a rigid shape. Thus, we define three speed classes as 'approach,' 'keep,' and 'far away,' which represent relative speeds of a car instance with respect to the capturing camera. If the distance between the camera and the instance is decreasing, the speed class is 'approach.' If the camera and the instance move in the same speed in the same direction, the class is 'keep.' Otherwise, the class is 'far away.' Last, we define four action classes for cars as 'go straight,' 'stop,' 'turn left,' and 'turn right.' Then, we manually assign those three attributes to each car instance. We use the KITTI object dataset [18], composed of 7,481 training images and 7,518 test images. Since only training images have bounding box annotations, we use those images. They contain 15,894 objects in total. We train the network in Figure 5 with 6,526 images with 13,894 objects. The test set consists of 955 images with 2,000 objects. Table 4 lists the classification accuracies. Note that the accuracies for cars are higher than those for pedestrians, since cars have less shape variations, as well as being rigid. Figure 12 presents examples of estimation results. In the topmost image, the proposed algorithm distinguishes the 'go straight' class from the 'stop' class correctly from the scene context.

Finally, we extend the proposed algorithm to estimate the FMs of animal instances (cats, dogs, and horses). As four-footed animals, they have similar motion characteristics, even though they do not belong to the same family. To construct the animal dataset, we collect frames, including cats, dogs, and horses, from YouTube [1]. Animals have eight direction classes and three speed classes in the same way as pedestrians. We do not perform the action classification for animals. We collect 5,516 images, including 6,626 animals: 5,302 animals are used for training and 1,324 for test. Table 4 also lists the classification accuracies for animals, where 'Baseline' denotes the results using the multi-class classification. All performances are improved due to the ordinal regression. In terms of the speed classification, animals are more challenging than pedestrians, since the differences between the 'slow' and 'fast' classes are often ambiguous in still images for animals. Figure 13 shows correct classification results, as well as failure cases. For example, in the third image, the direction prediction fails since the horse is slanted abnormally to turn fast.

6. Conclusions

We proposed a novel single-image FM estimation algorithm at the instance level. Using the MCP layer, the proposed algorithm extracts object and global context features for faithful FM estimation. The proposed algorithm performs three classification tasks to determine the future direction, speed, and action of an instance. Especially, we proposed the COR scheme for the ordinal regression of future direction. Experimental results demonstrated that the proposed algorithm yields reliable FM estimation performance and can be used for single and multi object tracking. Also, the proposed algorithm can be used for estimating the FMs of cars, cats, dogs, horses, as well as pedestrians.

Acknowledgements

This work was supported by 'The Cross-Ministry Giga KOREA Project' grant funded by the Korea government (MSIT) (No. GK18P0200, Development of 4D reconstruction and dynamic deformable action model based hyper-realistic service technology), by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2018R1A2B3003896), and by NAVER LABS.

References

[1] http://youtube.com/. 3, 8

[2] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *AVSS*, 2017. 7

[3] Kuang-Yu Chang, Chu-Song Chen, and Yi-Ping Hung. Ordinal hyperplanes ranker with cost sensitivities for age estimation. In *CVPR*, 2011. 3

[4] Yu-Wei Chao, Jimei Yang, Brian Price, Scott Cohen, and Jia Deng. Forecasting human dynamics from static images. In *CVPR*, 2017. 2

[5] Wei Chu and Zoubin Ghahramani. Gaussian processes for ordinal regression. *J. Mach. Learn. Res.*, 6:1019–1041, 2005. 2

[6] Wei Chu and S. Sathiya Keerthi. Support vector ordinal regression. *Neural Comput.*, 19(3):792–815, 2007. 2

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 3

[8] Koby Crammer and Yoram Singer. Online ranking by projecting. *Neural Comput.*, 17(1):145–175, 2005. 2

[9] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 5, 6

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 5, 6

[11] Dieter Devlaminck, Willem Waegeman, Bruno Bauwens, Bart Wyns, Patrick Santens, and Georges Otte. From circular ordinal regression to multilabel classification. In *ECML Workshop*, 2010. 3, 4

[12] Piotr Dollár, Ron Appel, Serge Belongie, and Pietro Perona. Fast feature pyramids for object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(8):1532–1545, 2014. 5, 6

[13] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(4):743–761, 2012. 3

[14] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley & Sons, 2001. 5

[15] Eibe Frank and Mark Hall. A simple approach to ordinal classification. In *ECML*, 2001. 2

[16] Ruohan Gao, Bo Xiong, and Kristen Grauman. Im2Flow: Motion hallucination from static images for action recognition. In *CVPR*, 2018. 1, 2, 6

[17] Kirill Gavrilyuk, Amir Ghodrati, Zhenyang Li, and Cees G. M. Snoek. Actor and action video segmentation from a sentence. In *CVPR*, 2018. 1

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *CVPR*, 2012. 8

[19] Pedro Antonio Gutiérrez, María Pérez-Ortiz, Javier Sánchez-Monedero, Francisco Fernández-Navarro, and César Hervás-Martínez. Ordinal regression methods: Survey and experimental study. *IEEE Trans. Knowl. Data. Eng.*, 28(1):127–146, 2016. 2

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 1

[21] Karel Hrbacek and Thomas Jech. *Introduction to Set Theory*. Dekker, 1984. 2

[22] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 1, 2, 4, 5, 6

[23] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, 2018. 1

[24] Won-Dong Jang and Chang-Su Kim. Interactive image segmentation via backpropagating refinement scheme. In *CVPR*, 2019. 1

[25] Han-Ul Kim and Chang-Su Kim. CDT: Cooperative detection and tracking for tracing multiple objects in video sequences. In *ECCV*, 2016. 2, 6, 7

[26] Kris M. Kitani, Brian D. Ziebart, James Andrew Bagnell, and Martial Hebert. Activity forecasting. In *ECCV*, 2012. 1, 2

[27] Matej Kristan, Jiri Matas, Aleš Leonardis, Michael Felsberg, Luka Čehovin, Gustavo Fernández, Tomáš Vojí r, Gustav Häger, et al. The visual object tracking VOT2015 challenge results. In *ICCVW*, 2015. 6

[28] Matej Kristan, Roman Pflugfelder, Aleš Leonardis, Jiri Matas, Luka Čehovin, Georg Nebehay, T. Vojíř, Gustavo Fernández, et al. The visual object tracking VOT2014 challenge results. In *ECCVW*, 2014. 6

[29] Jae-Han Lee and Chang-Su Kim. Monocular depth estimation using relative depth maps. In *CVPR*, 2019. 1

[30] Jun-Tae Lee and Chang-Su Kim. Image aesthetic assessment based on pairwise comparison – a unified approach to score regression, binary classification, and personalization. In *ICCV*, 2019. 1

[31] Jun-Tae Lee, Han-Ul Kim, Chul Lee, and Chang-Su Kim. Semantic line detection and its applications. In *ICCV*, 2017. 1

[32] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *ICCV*, 2011. 7

[33] Ling Li and Hsuan-Tien Lin. Ordinal regression by extended binary classification. In *NIPS*, 2007. 2, 4

[34] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Flow-grounded spatial-temporal video prediction from still images. In *ECCV*, 2018. 2

[35] Pengpeng Liang, Erik Blasch, and Haibin Ling. Encoding color information for visual tracking: Algorithms and benchmark. *IEEE Trans. Image Process.*, 24(12):5630–5644, 2015. 6

[36] Wen Liu, Weixin Luo, Dongze Lian, and Shenghua Gao. Future frame prediction for anomaly detection – a new baseline. In *CVPR*, 2018. 1

[37] Wei-Chiu Ma, De-An Huang, Namhoon Lee, and Kris M. Kitani. Forecasting interactive dynamics of pedestrians with fictitious play. In *CVPR*, 2017. 1, 2

[38] Simon Meister, Junhwa Hur, and Stefan Roth. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, 2018. 1

[39] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. MOT16: A benchmark for multi-object tracking. *arXiv, 1603.00831*, 2016. 7

[40] Roozbeh Mottaghi, Hessam Bagherinezhad, Mohammad Rastegari, and Ali Farhadi. Newtonian image understanding: Unfolding the dynamics of objects in static images. In *CVPR*, 2016. 1, 2

[41] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016. 2, 6, 7

[42] Zhenxing Niu, Mo Zhou, Le Wang, Xinbo Gao, and Gang Hua. Ordinal regression with multiple output cnn for age estimation. In *CVPR*, 2016. 3, 5

[43] Silvia L. Pintea, Jan C. van Gemert, and Arnold W.M. Smeulders. Déjà Vu: Motion prediction in static images. In *ECCV*, 2014. 2

[44] Joseph Redmon and Ali Farhadi. YOLOv3: an incremental improvement. *arXiv, 1804.02767*, 2018. 3

[45] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 6

[46] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 1

[47] Jacob Walker, Abhinav Gupta, and Martial Hebert. Patch to the future: Unsupervised visual prediction. In *CVPR*, 2014. 2

[48] Jacob Walker, Abhinav Gupta, and Martial Hebert. Dense optical flow prediction from a static image. In *ICCV*, 2015. 2

[49] Jacob Walker, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *ECCV*, 2016. 2

[50] Chao-Yuan Wu, Nayan Singhal, and Philipp Krähenbühl. Video compression through image interpolation. In *ECCV*, 2018. 1

[51] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(9):1834–1848, 2015. 6

[52] Takuma Yagi, Karttikeya Mangalam, Ryo Yonetani, and Yoichi Sato. Future person localization in first-person videos. In *CVPR*, 2018. 1, 2

[53] Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. DenseASPP for semantic segmentation in street scenes. In *CVPR*, 2018. 1

[54] Shanshan Zhang, Rodrigo Benenson, and Bernt Schiele. CityPersons: A diverse dataset for pedestrian detection. In *CVPR*, 2017. 3

[55] Yizhou Zhou, Xiaoyan Sun, Zheng-Jun Zha, and Wenjun Zeng. MiCT: Mixed 3D/2D convolutional tube for human action recognition. In *CVPR*, 2018. 1