PREDICTION-BASED LOAD CONTROL AND BALANCING FOR FEATURE EXTRACTION IN VISUAL SENSOR NETWORKS

Emil Eriksson, György Dán and Viktoria Fodor

School of Electrical Engineering, KTH Royal Institute of Technology, Stockholm, Sweden

ABSTRACT

We consider controlling and balancing the processing load in a visual sensor network (VSN) used for detecting local features, such as BRISK. We formulate a prediction problem with random missing data, and propose two regression-based algorithms for data reconstruction. Numerical results illustrate the performance of the proposed algorithms, and show that backward regression combined with the last value predictor can be used for controlling and balancing the processing load in VSNs with good performance.

1. INTRODUCTION

Computer vision has a wide range of applications in industry and in society, including the supervision of manufacturing, automated surveillance, remote medical diagnosis, navigation of automotives, or augmented reality. Therefore, affordable and reliable computer vision systems could become important building blocks in the emerging networked society [1, 2].

Computer vision systems today fall into one of two categories. In the first category expensive smart cameras are used that are capable of performing complex computer vision tasks locally, and can communicate the results to a central computer. In the second category cheap low complexity cameras capture the visual information from different viewpoints, then compress and transmit this information to a central server, which performs the visual analysis. As a consequence, the transmission bandwidth requirements to the server are significant, which limits the feasibility of the approach.

Visual sensor networks (VSNs) with forwarding nodes that are capable of processing visual information may provide a solution to perform visual analysis tasks at low cost, low transmission requirements and low delay. The processing delay can be decreased via parallel processing at the nodes. Furthermore, the transmission bandwidth requirements can be lower, since the in-network processing avoids the need of transmitting pixel information to the server. To achieve these gains, however, the VSN has to be able to control the workload and balance the workload distribution among the network nodes, which is a challenging task considering the high variability of the image content [3].

In this paper we propose to utilize the temporal correlation in video sequences to control and balance the load in the VSN. Specifically, we consider visual analysis based on local feature descriptors, where the processing load depends on the descriptor detection threshold. The processing load is distributed by allocating sub-areas of the images to the processing nodes. The balance of the processing load depends on the number of detected descriptors within the sub-areas.

Our aim is to predict the detection threshold that results in the desired number of detected features and the cut-points of the sub-areas that lead to a balanced load distribution. The information available from the processing of preceding images is limited: only the predicted parameter values and the resulting error in the number and in the balance of descriptors can be observed. Therefore, for each image we first have to estimate the threshold and cut-point values that should have been used for the preceding image, and then predict the optimal values of these parameters via autoregressive models.

Related to ours are recent works on visual analysis of video sequences in smart camera networks. The temporal correlation of video sequences is traditionally used to decrease the image information to be stored or transmitted, by applying inter-frame coding [4]. A similar idea, differential coding among the descriptors detected in the consecutive images of the video sequence, was used in [5] to decrease the bandwidth needed for transmitting the descriptors from smart cameras.

Our work is motivated by recent works on visual analysis in sensor networks [6, 7, 3]. In [6] the authors showed that the processing delay and the energy consumption increases linearly with the number of detected interest points, and consequently the control of this parameter is necessary. [7] demonstrated that centralized processing leads to significant delays in a VSN, and thus by distributing the processing load the performance of the VSN could be improved significantly. Statistical analysis of a large public image database revealed that the number and the spatial distribution of the descriptors have high variability and depend significantly on the image content [3]. Thus, the utilization of the temporal correlation in the video sequence is necessary for the efficient control of the visual analysis parameters.

The project GreenEyes acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number:296676.

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

2. SYSTEM MODEL AND PROBLEM STATEMENT

We consider a visual sensor network (VSN) that consists of a camera node C, a set of processing nodes $\mathcal{P}$, $|\mathcal{P}| = P$, and a sink node S. The camera node C produces a finite sequence $\{Z_i\}$ of images, numbered $i = 0, 1, \ldots, I$. It sends each image to the processing nodes, which detect and extract local descriptors, such as the recently proposed SURF [8] or BRISK [9] descriptors.

Interest point detection can be done using a blob detector, as in the case of SURF, or using an edge detector, as in the case of BRISK, and is parametrized by a detection threshold $\vartheta \in \mathbf{\Theta} \subseteq \mathbb{R}^+$. A point in an image is identified as an interest point if the detection scheme assigns a score higher than ϑ to it. The number of interest points detected in image i is a left continuous, decreasing, non-negative integer valued step function $f_i(\vartheta)$ of the detection threshold ϑ used by the interest point detection algorithm. Let us define the inverse $f_i^{-1} : \mathbb{N} \to \mathbf{\Theta}$ of f as $f_i^{-1}(m) = \max\{\vartheta | f_i(\vartheta) = m\}$. The maximum exists because f_i is a left continuous decreasing step function. We use the notation $\boldsymbol{\vartheta} = (\vartheta_1, \ldots, \vartheta_I)$ for the vector of thresholds used for the sequence of images.

In order to distribute the workload among the processing nodes in $\mathcal{P}$, the camera node creates P sub-areas of image i, one per processing node, and assigns sub-area $Z_{i,p}$ to processing node p. This scheme is referred to as area-split in [3]. We consider that the sub-areas are slices of the image, and for simplicity we assume slices are formed along the horizontal axis. The sub-areas in image i are thus determined by the cut-point location vector $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,P-1})$. We denote the set of feasible cut-point location vectors by $\mathcal{X}$. The number of interest points in sub-area $Z_{i,p}$ is a function $f_{i,p}(\vartheta, \boldsymbol{x}_i)$ of the detection threshold and of the cut-point location vector (in fact, of two of its components $x_{i,p-1}$ and $x_{i,p}$). The number of interest points in sub-area $Z_{i,p}$ determines the processing time and the energy consumption of node p.

Once the processing nodes are done with the detection and extraction for image i with the parameters $(\vartheta_i, \boldsymbol{x}_i)$, they send the descriptors to the sink node S. The sink uses the descriptors for a visual analysis task, such as object recognition and tracking, and it requires M^* descriptors per image. Since the score of each interest point is available at the sink, the sink can compute $(f_{i,1}(\vartheta, \boldsymbol{x}), \ldots, f_{i,P}(\vartheta, \boldsymbol{x}))$ for an arbitrary cut-point location vector $\boldsymbol{x}$ but only for $\vartheta \geq \vartheta_i$. Furthermore, it can compute $f_i(\vartheta)$ for any $\vartheta \geq \vartheta_i$. The sink can not, however, compute $f_i(\vartheta)$ for $\vartheta < \vartheta_i$, as it does not have access to image i. Consequently, if $f_i(\vartheta_i) < M^*$ then $f_i^{-1}(M^*)$ is not known. We denote by Υ_i the data available to the sink node about image i, and by Υ_{i-} the data available up to but not including image i.

2.1. Problem Formulation

The goal of the VSN is to detect the desired M^* number of interest points in every image in a balanced way, that is, each processing node should detect and extract $N^* \triangleq M^*/P$ interest points. Typically $M^* >> P$, thus it is reasonable to assume that M^* is divisible by P. Given $(\vartheta_i, \boldsymbol{x}_i)$, we define the error due to the mismatch in the number of detected interest points in image i as

$$e_i^D(\vartheta_i) = \left(f_i(\vartheta_i) - M^*\right)^2, \tag{1}$$

and the mean square error $e^D(\boldsymbol{\vartheta}) = \frac{1}{I}\sum_{i=1}^{I} e_i^D(\vartheta_i)$. Similarly, the error due to the lack of balance in the load of the processing nodes for image i as

$$e_i^B(\vartheta_i, \boldsymbol{x}_i) = \sum_{p=1}^{P} \left(f_{i,p}(\vartheta_i, \boldsymbol{x}_i) - N^*\right)^2, \tag{2}$$

and the mean square error $e^B = \frac{1}{I}\sum_{i=1}^{I} e_i^B(\vartheta_i, \boldsymbol{x}_i)$. Observe that $e_i^B(\vartheta_i, \boldsymbol{x}_i) \geq e_i^D(\vartheta_i)$, and thus minimizing (2) requires the minimization of (1). Since the functions f_i and $f_{i,p}$ are not known a priori, the problem is to find a predictor $\tau^*(\Upsilon)$ that minimizes the mean square error

$$\tau^* \in \arg\min_{\tau} \frac{1}{I}\sum_{i=1}^{I} e_i^D(\tau(\Upsilon_{i-})), \tag{3}$$

and a predictor $\gamma^*(\Upsilon)$ that minimizes the mean square error

$$\gamma^* \in \arg\min_{\gamma} \frac{1}{I}\sum_{i=1}^{I} e_i^B(\tau^*(\Upsilon_{i-}), \gamma(\Upsilon_{i-})). \tag{4}$$

The solution of the prediction problems (3) and (4) using standard predictors is not immediate for two reasons. First, the sets of minimizers $\theta_i^* = \{\vartheta | e_i^D(\vartheta) = 0\}$ and $\Xi_i^* = \{\boldsymbol{x} | e_i^B(\vartheta^*, \boldsymbol{x}) = 0, \vartheta^* \in \theta_i^*\}$ need not be singletons, because $f_i(\vartheta)$ and $f_{i,p}(\vartheta, \boldsymbol{x})$ and thus (1) and (2) are step functions in ϑ and in $\boldsymbol{x}$. Second, if $f_i(\vartheta_i) < M^*$ then θ_i^* is unknown, and can thus not be used for prediction. In the following we propose two regression-based solutions to address the second problem.

3. REGRESSION-BASED RECONSTRUCTION

We propose two regression-based methods for estimating θ_i^*, which allows us to evaluate various autoregressive models that require ϑ_i^* for prediction.

Consider an image i for which the predicted detection threshold $\hat{\vartheta}_i$ results in $f_i(\hat{\vartheta}_i) < M^*$. The goal is to estimate some $\hat{\vartheta}_i^* \in \theta_i^*$ that can be used to predict $\hat{\vartheta}_{i+1} \in \theta_{i+1}^*$. The key tenet of the proposed approach is to use preceding images for which $f_j(\hat{\vartheta}_j) \geq M^*$ for estimating the slope of the function f_i^{-1} around M^*. Let $\mathcal{I}_{i-}$ be the set of indices of the images before image i for which the estimated detection threshold $\hat{\vartheta}_j$ resulted in more than M^* interest points, i.e., $f_j(\hat{\vartheta}_j) \geq M^* \; \forall j \in \mathcal{I}_{i-}$. We can use the images in $\mathcal{I}_{i-}$ to estimate the slope of the function f_i^{-1} around M^* in two ways: in the forward direction and in the backward direction.

Forward estimate: To obtain the forward estimate of the slope of the function we define the forward regression coefficient

$$\beta_{i-}^{f} = \frac{\frac{1}{|\mathcal{I}_{i-}|}\sum_{j\in\mathcal{I}_{i-}}(f_j(\hat{\vartheta}_j) - M^*)(\hat{\vartheta}_j - f_j^{-1}(M^*))}{\frac{1}{|\mathcal{I}_{i-}|}\sum_{j\in\mathcal{I}_{i-}}(f_j(\hat{\vartheta}_j) - M^*)^2}, \quad (5)$$

which is the estimated slope of the piece-wise linear extension of f_i^{-1} in the forward direction (i.e., beyond M^*). We then use the forward regression coefficient to obtain the estimated threshold

$$\hat{\vartheta}_i^{f*} = \hat{\vartheta}_i - (f_i(\hat{\vartheta}_i) - M^*)\beta_{i-}^{f} \quad (6)$$

Backward estimate: To obtain the backward estimate of the function's slope we use the same linear regression but in the backward direction. In the backward direction (i.e., less than M^* interest points) we can compute the regression for arbitrary difference $d < M^*$ based on the available data Υ_{i-}. For a particular difference d after simplification we obtain

$$\beta_{i-}^{b}(d) = \frac{1}{|\mathcal{I}_{i-}|}\sum_{j\in\mathcal{I}_{i-}}\frac{f_j^{-1}(M^*) - f_j^{-1}(M^* - d)}{d}, \quad (7)$$

which is the average backward difference quotient of f^{-1} at M^* over the images in $\mathcal{I}_{i-}$. Using the backward regression coefficient we obtain the estimated threshold

$$\hat{\vartheta}_i^{b*} = \hat{\vartheta}_i - (f_i(\hat{\vartheta}_i) - M^*)\beta_{i-}^{b}. \quad (8)$$

Proposition 1. *Assume that for every d the backward difference quotient $\frac{f_i^{-1}(M^*) - f_i^{-1}(M^*-d)}{d}$ of f_i^{-1} at M^* can be modeled by an i.i.d. random variable, and is independent of $f_i^{-1}(M^*)$. Then the estimated threshold $\hat{\vartheta}_i^{b*} = \arg\min_{\vartheta} E[e_i^D(\vartheta)]$.*

Proof. Since the backward difference quotient is independent of $f_i^{-1}(M^*)$, the backward difference quotient of images $j \in \mathcal{I}_{i-}$ is an unbiased sample of that of all images $j < i$. Since $\beta_{i-}^{b}(d)$ is the sample mean of the backward difference quotient, it is the minimum variance unbiased estimator, and thus it minimizes the expected square error. □

Given $\hat{\vartheta}_i^{b*}$ or $\hat{\vartheta}_i^{f*}$ we can use a similar regression for estimating the cut-point location vector $\hat{\boldsymbol{x}}_i$ that would minimize $e_i^B(\hat{\vartheta}_i^*, \boldsymbol{x}_i)$.

4. PERFORMANCE EVALUATION

We use two video traces to evaluate the proposed regression-based prediction and to compare the performance of different predictors. The traces are surveillance recordings from two different locations, and have a resolution of 1920×1080 pixels at 25 frames per second. One trace, referred to as the "Pedestrian" trace, consists of 375 frames and shows a pedestrian intersection with people moving from one side to the other, covering up and uncovering interest points as people move. The second trace, referred to as the "Rush hour" trace, contains 473 frames and shows a busy road with slow moving vehicles. The line of vision is parallel to the street, meaning vehicles move mostly towards and away from the camera. The more chaotic movements of the Pedestrian trace sometimes produce large changes in both θ^* and Ξ^*. The VSN we consider contains two processing nodes, and we use BRISK for interest point detection with $M^* = 400$.

As a basis of comparison for the proposed regression based reconstruction we use two methods. The first method, referred to as the *Scaling* method, scales the predicted threshold $\hat{\vartheta}_i$ by a constant factor α, $0 < \alpha < 1$ whenever $f_i(\hat{\vartheta}_i) < M^*$, i.e., the scaled value of the threshold prediction for image i is $\hat{\vartheta}_i^{S*} = \alpha \cdot \hat{\vartheta}_i$. Off-line evaluation shows that the scaling method produces best results for $\alpha = 0.98$. The second method, referred to as the *Clairvoyant* scheme, has knowledge of $\hat{\vartheta}_i^* = \vartheta_i^* = f_i^{-1}(M^*)$ and $\hat{x}_i^* = x_i^* = \frac{\max(\Xi_i^*) + \min(\Xi_i^*)}{2}$, and can use it for predicting $\hat{\vartheta}_{i+1}$ and $\hat{x}_{i+1}$.

The predictions of both the threshold and the cut-point values are done using autoregressive (AR) prediction models of orders 1, 2, and 10. Our choice of prediction models and their orders are based on the partial autocorrelation function of ϑ_i^* and x_i^* for the two video traces. The AR predictors are initially trained using data from the first quarter of the trace, and then retrained after each frame. Alongside the AR predictors we use the last value predictor (denoted by $Y(i-1)$), which predicts $\hat{\vartheta}_{i+1} = \hat{\vartheta}_i^*$.

In Figure 1 we compare the mean square error (MSE) of the four reconstruction schemes. Subfigure 1(a) was obtained by using the last value predictor for predicting $\hat{\vartheta}_i$; whenever $f_i(\hat{\vartheta}_i) < M^*$ we use one of three reconstruction schemes to reconstruct $\hat{\vartheta}_i^*$ and compute the resulting squared error $(\hat{\vartheta}_i^* - \vartheta_i^*)^2$. To avoid the propagation of the prediction error, we use ϑ_i^* instead of $\hat{\vartheta}_i^*$ as input for predicting $\hat{\vartheta}_{i+1}$. The values plotted are the MSE of the backward regression scheme, i.e., $(\hat{\vartheta}_i^{b*} - \vartheta_i^*)^2$, divided by the MSE of the three schemes. The results show that backward regression performs best, in accordance with Proposition 1.

Subfigure 1(b) shows the MSE of the last value predictor combined with the *Clairvoyant* scheme, divided by the MSE of the predicted thresholds for four predictors combined with three reconstruction schemes. The results were obtained by using the reconstructed thresholds $\hat{\vartheta}_i^*$ for predicting the subsequent thresholds $\hat{\vartheta}_j$, $j > i$. This creates a feedback loop where the choice of predictor influences the frames for which reconstruction is needed, and reconstruction influences the prediction performance. From the figure we see that the backward regression scheme has a slight advantage over the forward regression scheme in almost all scenarios, and they both greatly outperform the *scaling* scheme. Therefore from this point on we only consider the backward regression scheme for threshold reconstruction.

Figure 2 shows the MSE of the predicted threshold values

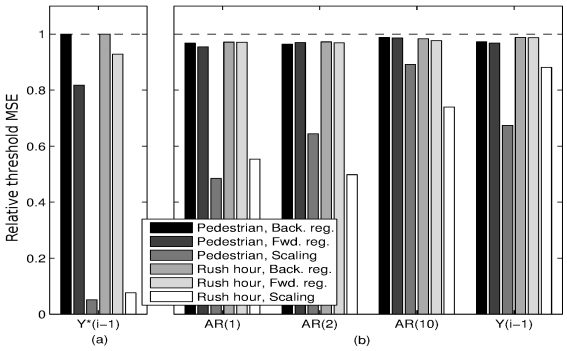

Fig. 1. Relative MSE of (a) the reconstructed thresholds (predictors trained with *Clairvoyant* scheme) and (b) the predicted thresholds (predictors trained with reconstructed thresholds).

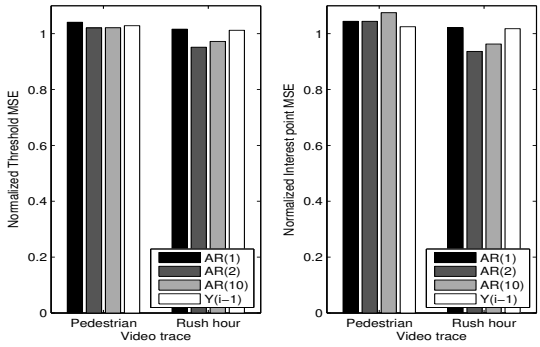

Fig. 2. MSE of threshold prediction normalized by that of the *Clairvoyant* last value predictor.

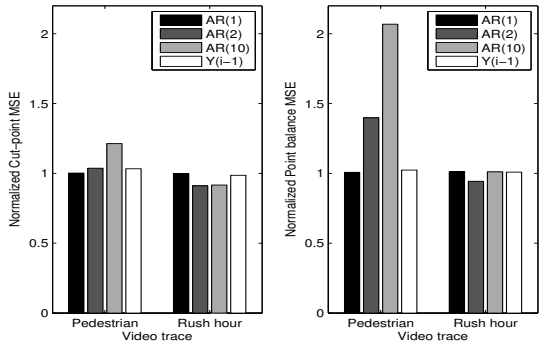

Fig. 3. MSE of cut point prediction normalized by that of the *Clairvoyant* last value predictor.

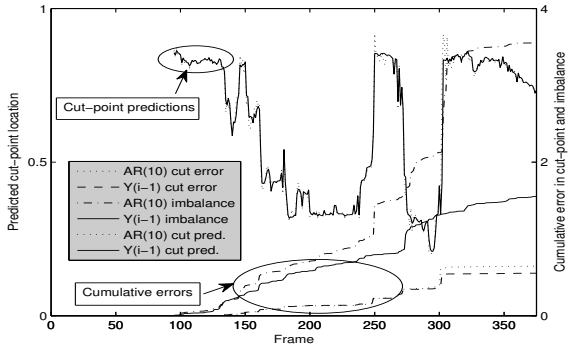

Fig. 4. Evolution of predicted cut-point and MSE for the Pedestrian trace.

$([\hat{\vartheta}_i - \vartheta_i^*]^2)$ and of the number of interest points detected (e_i^D) for the different prediction models, normalized by that of the last value predictor with the *Clairvoyant* scheme. The figure shows that the gains of using a higher order predictor are small, and in certain cases a high order predictor may even perform worse than the last value predictor. Finally, compared to a static optimal scheme that uses $\hat{\vartheta}_i = \hat{\vartheta}$ that minimizes e^D, we found that prediction reduces the MSE of detected interest points by a factor of 5 to 25.

Figure 3 shows corresponding MSE results for cut-point location $([\hat{\boldsymbol{x}}_i - \boldsymbol{x}_i^*]^2)$ and the point balance (e_i^B) normalized by that of the last value predictor with the *Clairvoyant* scheme. For the Pedestrian trace the normalized point balance MSE of the AR(10) predictor is roughly twice that of the AR(1) and of the last value predictor. While it may sound counter-intuitive that higher order predictors perform worse in terms of point balance, we have to remember that what we use to train the predictors is the cut-point location, and not the point balance.

In Figure 4, we observe what happens when there are sudden changes in the cut-point location. We can see that the error in cut-point prediction is not that sensitive to sudden changes, and the errors of the AR(10) and the last value prediction models are fairly similar. However, after the cut-point suddenly changes, it takes the AR(10) predictor some time to settle around the new cut-point. During the settling time, the point balance error grows quickly, indicating that there is a high concentration of interest points around the cut-point. We can therefore conclude that although the two error measures are related, the distribution of interest points in a frame can cause a predictor with higher settling time to produce large errors in point balance. This in turn explains, why a low order AR predictor can perform better than higher order ones.

5. CONCLUSION

We considered the problem of controlling and balancing the processing load of detecting local visual features in a visual sensor network. We proposed a backward and a forward regression-based algorithm for reconstructing missing data. We used these data reconstruction algorithms in conjunction with a variety of predictors for predicting the detection threshold and the cut-point location. Our numerical results show that, in accordance with the analytical results, backward regression based reconstruction performs best. Furthermore, the simple last value predictor proves to achieve consistently good performance both for detection threshold and cut-point location prediction. Combined with low computational complexity, its good performance makes the last value predictor a good candidate for load control and balancing in VSNs.

6. REFERENCES

[1] M. Cesana, A. Redondi, N. Tiglao, A. Grilo, J. Barcelo-Ordinas, M. Alaei, and P. Todorova, "Real-time multimedia monitoring in large-scale wireless multimedia sensor networks: Research challenges," in *8th EURO-NGI Conference on Next Generation Internet (NGI)*, 2012.

[2] A. Marcus and O. Marques, "An eye on visual sensor networks," *IEEE Potentials*, vol. 31, no. 2, pp. 38 –43, Apr 2012.

[3] M. A. Khan, G. Dan, and V. Fodor, "Characterization of surf interest point distribution for visual processing in sensor networks," in *Proc. of 18th International Conference on Digital Signal Processing (DSP)*, 2013.

[4] G. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[5] L. Baroffio, M. Cesana, A. Redondi, S. Tubaro, and M. Tagliasacchi, "Coding video sequences of visual features," in *Proc. of IEEE Intl. Conf. on Image Processing (ICIP)*, 2013.

[6] A. Redondi, M. Cesana, and M. Tagliasacchi, "Rate-accuracy optimization in visual wireless sensor networks," in *Prof. of IEEE ICIP*, 2012.

[7] A. Redondi, L. Baroffio, A. Canclini, M. Cesana, and M. Tagliasacchi, "A visual sensor network for object recognition: Testbed realization," in *Proc. of 18th Intl. Conference on Digital Signal Processing (DSP)*, 2013.

[8] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (surf)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346 – 359, 2008.

[9] S. Leutenegger, M. Chli, and R. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *Proc. of IEEE ICCV*, 2011.