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

Emil Eriksson, György Dán and Viktória 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.
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 \( P \), \(|P| = P\), 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 \( \vartheta \in \Theta \subseteq \mathbb{R}^+ \). A point in an image is identified as an interest point if the detection scheme assigns a score higher than \( \vartheta \) in 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 \( \vartheta \) used by the interest point detection algorithm. Let us define the inverse \( f_i^{-1} : \mathbb{N} \to \Theta \) 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 \( \vartheta = (\vartheta_1, \ldots, \vartheta_J) \) for the vector of thresholds used for the sequence of images.

In order to distribute the workload among the processing nodes in \( P \), 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 \( x_i = (x_{i,1}, \ldots, x_{i,p-1}) \). 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 \( f_{i,p}(\vartheta, x_i) \) 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 \( (\vartheta_i, x_i) \), 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 key tenet of the proposed approach is to use preceding images for which \( f_i(\vartheta_i) \geq M^* \) for estimating the slope of the function \( f_i^{-1} \) around \( M^* \). Let \( \mathcal{I}_- \) be the set of indices of the images before image \( i \) for which the estimated detection threshold \( \vartheta_j \) resulted in more than \( M^* \) interest points, i.e., \( f_j(\vartheta_j) \geq M^* \). We can use the images in \( \mathcal{I}_- \) to estimate the slope of the function \( f_i^{-1} \) around \( M^* \) in two ways: in the forward direction and in the backward direction.

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 \( N^* = \frac{M^*}{P} \) interest points. Typically \( M^* > P \), thus it is reasonable to assume that \( M^* \) is divisible by \( P \). Given \( (\vartheta_i, x_i) \), we define the error due to the mismatch in the number of detected interest points in image \( i \) as

\[
e^D_i(\vartheta_i) = (f_i(\vartheta_i) - M^*)^2,
\]

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

\[
e^B_i(\vartheta_i, x_i) = \sum_{p=1}^P (f_{i,p}(\vartheta_i, x_i) - N^*)^2,
\]

and the mean square error \( e^B = \frac{1}{I} \sum_{i=1}^I e^B_i(\vartheta_i, x_i) \).

The solution of the prediction problems (3) and (4) using standard predictors is not immediate for two reasons. First, the sets of minimizers \( \vartheta^*_i = \{\vartheta | e^D_i(\vartheta) = 0\} \) and \( \Xi^*_i = \{x | e^B_i(\vartheta^*_i, x) = 0, \vartheta^*_i \in \vartheta^*_i\} \) need not be singletons, because \( f_i(\vartheta) \) and \( f_{i,p}(\vartheta, x) \) are step functions in \( \vartheta \) and in \( x \). Second, if \( f_i(\vartheta_i) < M^* \) then \( \vartheta^*_i \) 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 \( \vartheta^*_i \), which allows us to evaluate various autoregressive models that require \( \vartheta^*_i \) 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 \( \hat{\vartheta}^*_i \in \vartheta^*_i \) that can be used to predict \( \hat{\vartheta}_{i+1} \in \vartheta^*_{i+1} \). The key tenet of the proposed approach is to use preceding images for which \( f_j(\vartheta_j) \geq M^* \) for estimating the slope of the function \( f_i^{-1} \) around \( M^* \). Let \( \mathcal{I}_- \) be the set of indices of the images before image \( i \) for which the estimated detection threshold \( \vartheta_j \) resulted in more than \( M^* \) interest points, i.e., \( f_j(\vartheta_j) \geq M^* \). We can use the images in \( \mathcal{I}_- \) to estimate the slope of the function \( f_i^{-1} \) around \( M^* \) in two ways: in the forward direction and in the backward direction.

675
Forward estimate: To obtain the forward estimate of the slope of the function we define the forward regression coefficient

$$\beta_i^f = \frac{\sum_{j \in I_{i-}} (f_j(\hat{\theta}_j) - M^*) (\hat{\theta}_j - f_j^{-1}(M^*))}{|I_{i-}| \sum_{j \in I_{i-}} (f_j(\hat{\theta}_j) - M^*)^2}, \quad (5)$$

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

$$\hat{\theta}_i^f = \hat{\theta}_i - (f_i(\hat{\theta}_i) - M^*) \beta_i^f.$$  

(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 \( \mathbf{Y}_{i-} \).

For a particular difference \( d \) after simplification we obtain

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

which is the average backward difference quotient of \( f_j^{-1} \) at \( M^* \) over the images in \( I_{i-} \). Using the backward regression coefficient we obtain the estimated threshold

$$\hat{\theta}_i^b = \hat{\theta}_i - (f_i(\hat{\theta}_i) - M^*) \beta_i^b.$$  

(8)

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

Proof. Since the backward difference quotient is independent of \( f_j^{-1}(M^*) \), the backward difference quotient of images \( j \in I_{i-} \) 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. \( \Box \)

Given \( \hat{\theta}_i^{bs} \) or \( \hat{\theta}_i^{bs} \) we can use a similar regression for estimating the cut-point location vector \( \mathbf{x} \), that would minimize \( e_i^D(\hat{\theta}_i^b; \mathbf{x}_i) \).

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 \( \theta^* \) and \( \Xi^* \). 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 \( \hat{\theta}_i \) by a constant factor \( \alpha \), \( 0 < \alpha < 1 \) whenever \( f_i(\hat{\theta}_i) < M^* \), i.e., the scaled value of the threshold prediction for image \( i \) is \( \hat{\theta}_i^S = \alpha \cdot \hat{\theta}_i \). 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 \( \hat{\theta}_i^* = \hat{\theta}_i = f_i^{-1}(M^*) \) and \( \hat{x}_i^* = x_i^* = \frac{\max(\Xi^*) + \min(\Xi^*)}{2} \), and can use it for predicting \( \hat{\theta}_{i+1} \) 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 \( \hat{\theta}_i^* \) 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 \( \hat{\theta}_{i+1} = \hat{\theta}_i^* \).

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 \( \hat{\theta}_i \); whenever \( f_i(\hat{\theta}_i) < M^* \) we use one of three reconstruction schemes to reconstruct \( \hat{\theta}_i^* \) and compute the resulting squared error \( (\hat{\theta}_i^* - \hat{\theta}_i^*)^2 \). To avoid the propagation of the prediction error, we use \( \hat{\theta}_i^* \) instead of \( \hat{\theta}_i^* \) as input for predicting \( \hat{\theta}_{i+1} \). The values plotted are the MSE of the backward regression scheme, i.e., \( (\hat{\theta}_i^{bs} - \hat{\theta}_i^*)^2 \), 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 \( \hat{\theta}_i^* \) for predicting the subsequent thresholds \( \hat{\theta}_j, j > i \). 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.

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.

5. CONCLUSION
6. REFERENCES


