INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Graph Grammar Based Object Detecting and Tracking

Yuqing Song*, Dongpeng Yue, Shaoqing Mo

School of Automotive and Transportation, Tianjin University of Technology and Education, Tianjin, China

* *Corresponding author's Email: yqsong7@hotmail.com*

Abstract: In this paper we introduce a graph grammar based method to fuse the low level features and apply them to object detecting and tracking. In our algorithm, the graph grammar rules are used to detect the object in the beginning of the video sequence and then dynamically adjust the tracking procedure. Our tracking algorithm consists of two phases: key points tracking and tracking by graph grammar rules. The key points are computed by using salient level set components. All key points, as well as the colors and the tangent directions, are fed to a Kalman filter for object tracking. Then the graph grammar rules are used to dynamically examine and adjust the tracking procedure to make it robust. The effectiveness of the algorithm has been demonstrated by experiments.

Keywords: Object tracking; object detection; multi-feature fusion; graph grammar; semantics based tracking

1. Introduction

The explosive growth of digital video in recent years has created a need for effective and efficient video object detecting and tracking. Mastery of these technologies involves efforts in many fields including image processing, computer vision, and pattern recognition. Object detecting and tracking are important in vision-based applications, such as video surveillance, video retrieval, motion-based human identification, and video-based motion analysis and synthesis.

The general process of an object detecting and tracking algorithm includes detecting the qualifying target feature, estimating and updating the target state, predicting the movement of the target and getting into the next round of the tracking process. The detecting and tracking algorithms can be classified as bottom-up and top-down approaches. Without prior knowledge, the bottom-up approach directly computes the characteristics of the video target and tracks the target by tracking the computed features, such as [1]. The top-down approach uses prior knowledge to establish a dynamic target model, randomly generates a number of assumptions in the state space, and solves the posterior probability equation for object detection and tracking, such as the particle filter tracking algorithms [2, 3].

Many technologies are involved in object detecting and tracking. One key issue is the feature extraction and selection. Based on the feature selection and application, we divide the object detecting and tracking algorithms into the following categories: region-based, contour-based, model-based, and feature-based object detecting and tracking.

For region-based algorithms [4, 5, 6], the object information is implicit in the prior region. The shape of the region can be rectangular or irregular. The texture, color, or motion-based features in the region are extracted and used to track the target object. Contour-based algorithms [7, 8, 9] often use a deformable template. Initially the template is a closed curve drawn by the user. The curve may contract or expand under the external or internal energy. It gradually deforms into the real object boundary. In general, the methods of this category require an accurate initial value. Model-based methods [10, 11, 12] approximate the target ob-

ject by using a 2D/3D model. The models consist of geometrical representations of known objects, which can be placed in arbitrary positions and orientations. Model-based methods allow prior knowledge of the shape and appearance of specific objects to be used in detecting and tracking. Feature-based methods [13, 14, 15] take advantage of invariant features in movement, such as corner, texture, and color. The methods generally include two processes: feature extraction and feature matching. The major difference between feature-based and region-based methods is that the latter use the object as a whole, while the former use certain features of the object.

Research in object tracking has made considerable progress in the recent decade. However, due to a lack of deep understanding in human vision, especially in the mechanism of human intelligence, object tracking in a complex environment is still a big challenge for scientists to overcome. One of the key problems is how to effectively fuse the shape, texture, and color features and apply them to object detecting and tracking.

Image/video features have three levels, with the top level listed first: semantic level, visual level, and image processing level (pixel). Image processing level is the lowest level, where noise and distortion are reduced and certain low level features are analyzed. At the intermediate level, visual features are extracted. Typical visual features include color, texture, and shapes. The intermediate level is not domain-specific. It computes and stores the spatial or geometric properties of the color/intensity distribution. The visual features are provided to the semantic level, where domain knowledge is applied into object recognition and scene interpretation. The domain knowledge consists of descriptions of objects or entities in the domain. Figure 1 sketches a general video understanding process. The gap between the semantics of images/videos and the features automatically retrievable has driven researchers for new theories and technologies over decades. It is still a challenge today. The bottleneck is at the visual level. Specifically the following problem remains open at the visual level: how to fuse the visual elements into objects and scenes.

In video object detection and tracking process, the target object is dynamic, and its shape and status change with time. Graph grammar method is more suitable than other formal tools to describe the dynamic characteristics.

As a tool for graph transformation, graph grammar method [16, 17] has a history over 30 years, and there

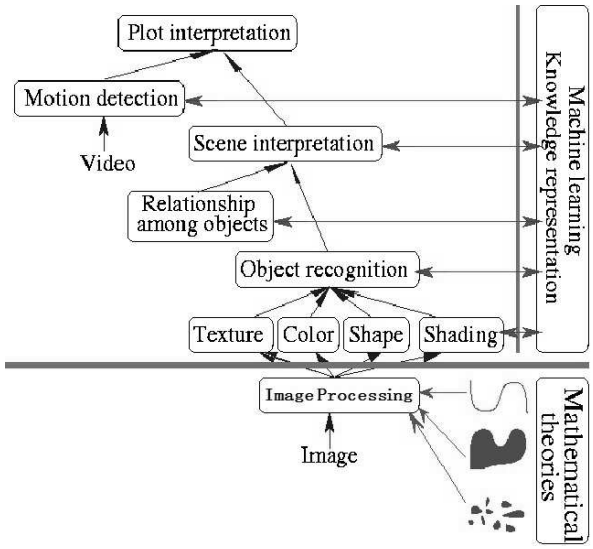


Figure 1 The sketch of video understanding process

are a lot of applications [18, 19, 20]. Since Chomsky first established a formal language system in 1956, formal language theory has made a considerable progress with a profound impact on the computer sciences and human society. With the rapid development of multimedia technology, charts and other visual data have been used extensively; Chomsky grammars are not qualified for the description and analysis of these two-dimensional objects. As an expansion of one- dimensional grammars, graph grammars can formally describe objects and their changes in the two-dimensional space, and provide theoretical and technical support for their definition, generation, and transformation. An earliest graph grammar was proposed to solve an image processing problem [21]. In 1969 Pfaltz and Rosenfeld published a paper entitled "Web Grammars" [22], opened a prelude for the study of graph grammars. Since then, computer scientists have explored extensively in both theories and applications, and achieved greatly in the formalization and implementation of various graph grammars.

When a human being observes a video frame, the frame is decomposed in the human brain into various visual elements according to the shape, texture, color, and other low-level features. Just as words make up phrases, which further make up sentences, the visual elements, as well as their interrelationships, make the objects and scenes, allowing human brain to quickly interpret the frame image. These elements and their mutual relations form a graph structure, and video understanding is a process of graph transformation. In this sense, graph grammar method is an appropriate tool for image and video understanding. Existing graph

grammars are mainly used in the identification and analysis of various charts, and applications on image/ video analysis are rarely reported. In this paper we propose a method based on graph grammar to fuse the low level features and apply them to video object detecting and tracking. In our algorithm, the graph grammar rules are used to detect the object in the beginning of the video sequence and then dynamically examine and adjust a Kalman filter based key point tracking procedure. Our algorithm is applied to license plate recognition and automatic facial feature points tracking. As we know, this is the first work on graph grammar based object detecting and tracking. It explores a new way for semantics based object detecting and tracking. The robustness and effectiveness of the algorithm has also been demonstrated by experiments.

This paper is an extension of our previous work [23]. The rest of the paper is organized as follows. In Section II we introduce our scheme for multi-feature fusion and template matching. In Section III the semantics based tracking method is elaborated. We report the experimental results in Section IV and conclude in Section V.

2. Multi-feature Fusion and Template Matching

A core step in object detecting and tracking is template matching. It is crucial to select proper features for template matching. Tracking by individual features, such as color or motion, is the main reason why most tracking algorithms are not as robust as expected. In order to better describe the object, multi-feature fusion is necessary. In our approach the feature fusion can be separated into two main stages. At the first stage, the level set components of a video frame are computed and the visual elements are selected; at the second stage, the visual elements are grouped into objects, based on the graph grammar rules. Figure 2 summarizes the main steps of feature fusion.

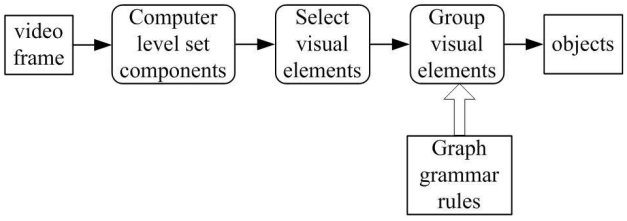


Figure 2 Feature fusion

Similar to a context free string grammar, a graph-grammar consists of a set of productions (rules) that can be used to construct or recognize valid sentences in a graph (network) language. For example, a face consists of a nose, a mouth, and two eyes with eyebrows. A rule for the object face is shown in Figure 3. In general, a rule of a graph grammar, expressed as $\alpha \leftarrow \beta$, has two parts:

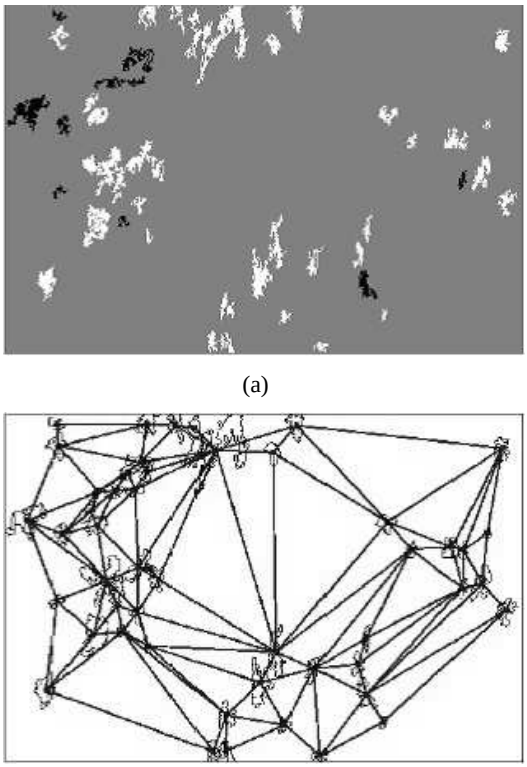
- α is a name or a label of an object;
- β is a sub-graph where a node is an element or a sub-object and an edge specifies the requirement for the relationship between the two connected nodes.

Figure 3 A rule for face recognition. The edges specify the required spatial relationships of the connected parts

In this paper we propose a multi-feature strategy which uses image level sets to investigate the chiaroscuro patterns and then fuse the low level features (shape, texture, color, etc.) by a graph grammar method to form an effective description of the object. In the process of graph transformation, the low level features are transformed into high-level semantics, using the domain knowledge embedded in the grammar rules.

In our algorithm, we generate the graph grammar rules by manual input or training from the sample images. For objects of regular shapes, such as a license plate, we manually specify the graph grammar rules. For objects of variant shapes, such as human faces, we need to train the grammar rules from sample images.

In template matching, an image or a video frame is first broken down into level set components, which are then selected based on their shapes, color, and other features. The selected components are the visual elements. We construct the Delaunay graph of the visual elements, as is shown in Figure 4. We use a graph matching scheme to match the sub-graph in a grammar rule against the visual element graph. Once a match is found, the matched visual elements constitute the target object we are looking for. In graph matching, we do not require an exact match to avoid the NP complexity of the exact matching process. We calculate the vertex compatibility matrix and edge compatibility matrix between the visual element graph and the sub-graph in a grammar rule, and measure the similarity between the two models, which serves as the



(a) (b)

Figure 4 (a) The visual elements, and (b) its Delaunay graph

objective function for optimally selecting vertex mapping matrix M between the two models. The matrix M is found by using Sinkhorn's alternative normalization method for M's rows and columns after M's elements are relaxed to be continuous [24].

2.1 License Plate Recognition

Automatic license plate recognition (LPR) plays a key role in intelligent transport systems. LPR is a powerful tool in numerous applications, including electronic toll and traffic management, commercial vehicle operations, motor vehicle law enforcement, origin destination survey, and security control of restricted areas. LPR consists of 3 stages: license plate location (LPL), license plate character segmentation (LPS), and optical character recognition (OCR). In the LPL stage, the region of the license plate is located in the input image; in the LPS stage, the system finds the individual characters on the plates; and in the third stage the characters are recognized.

Typical LPR techniques used in LPR systems include neural network [25], template matching [26], inductive learning [27], fuzzy Hough transform [28], and multi-feature [29]. LPR is a sophisticated task in a typical real-world scenario, where severe imaging conditions must be handled. An effective and robust LPR system must be able to compensate for all the variables that can affect the ability to produce an accurate recognition, such as time of day, weather, illumination wavelengths, and angles between the cameras and the license plates. Most existing LPR techniques and systems reduce the complexity by applying restrictions on the image conditions, such as designated routes, fixed illumination, limited vehicle speed, and stationary backgrounds.

In this paper a graph grammar based method is used for licensing plate recognition. In our approach, we first employ a level set based LPL process [30] to find candidate plate locations. Then a graph grammar is applied to each candidate location to recognize the plate. The grammar consists of 3 rules. The first two rules recognize individual characters using a bipolar shape matching scheme [31]. The third rule represents the layout of license plate by a graph. The plate recognition is achieved by a graph matching process.

In China, the standard license plate consists of 7 characters. The first one is a Chinese character which is an abbreviation of Chinese provinces. The second one is a letter ranging from A to Z except the letter I. The third and fourth ones are letters or numbers. The fifth to seventh ones are numbers ranging from 0 to 9.

After the plate localization is done, for each candidate plate location, we conduct plate recognition by using a graph grammar. The grammar can be simply written in BNF (Backus-Naur Form) as:

(r_1) <C_1> ← 京 | 津 | 苏

(r_2) <C_2> ← A | B | C | … | 0 | 1 | 2 | … | 9

(r_3) <Plate> ← <C_1> <C_2> <C_2> <C_2> <C_2> <C_2> <C_2>

The first two rules r_1 and r_2 are used for character recognition; the rule r_3 recognizes the license plate as a whole by using the characters recognized by r_1 and r_2, as well as their spatial relationship.

The character set of the standard Chinese license plate consists of less than 100 characters, each of which uses a standard font. For each character, we extract its standard shape as a 2D region, which is used in shape matching for character recognition. The character recognition process consists of the following steps. Firstly in each candidate plate location, we compute the level set components. Secondly for each component, we conduct a shape matching against the standard shape of each plate character, using a bipolar shape matching scheme [31]. Thirdly if a best matched character is found, then the character recog-

nition in this component is successful; otherwise it fails.

A discussion on the spatial relationship among characters is needed when using rule r_3 to recognize the license plate as a whole. We make a bounding rectangle for each character, as is shown in Figure 5. In a license plate, two adjacent characters x and y should meet the following spatial requirements:
(1) The widths of their bounding rectangles should be the same, i.e., $|W_x - W_y| \leq \varepsilon_1$, where $W_x(W_y)$ is the width of the bounding rectangle of $x(y)$, and ε_1 is the maximum allowable error.
(2) The heights of their bounding rectangles should be the same, i.e., $|H_x - H_y| \leq \varepsilon_2$, where $H_x(H_y)$ is the height of the bounding rectangle of $x(y)$, and ε_2 is the maximum allowable error.
(3) The top sides of the bounding rectangles should be parallel, i.e., $|\gamma_x - \gamma_y| \leq \varepsilon_3$, where $\gamma_x(\gamma_y)$ is the angle of the top side of the bounding rectangle of $x(y)$, and ε_3 is the maximum allowable error.
(4) The bottom sides of the bounding rectangles should be collinear.
(5) The distance between x and y should be in a required range.

Figure 5 The bounding rectangles of the plate characters

Based on the above discussion, the grammar rule r_3 can be expressed as a graph (see Figure 6), where C_1 and C_2 are the vertices, and the red lines between the adjacent vertices are the edges, specifying the spatial requirements of the vertices. The process to recognize the license plate as a whole is as follows. Firstly we organize all recognized characters and their spatial relationship by a graph, called the *character graph* of the candidate plate location. Secondly we use the rule r_3 to match the character graph. If a matched sub-graph is found, the matched sub-graph contains a real license plate.

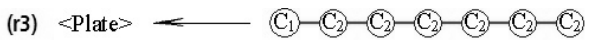


Figure 6 The grammar rule r_3 expressed by a graph

2.2 Facial Feature Points Detecting and Tracking

We apply the proposed method to automatic facial feature points detecting and tracking. Facial feature points are generally referred to as facial salient points. Commonly used methods for localizing these points can be divided into two categories: active shape modal (ASM) [32] and active appearance modal (AAM) [33]. Existing methods do not work well when facial expression and posture have complicated changes. As compared with the existing methods, our approach has the following two advantages. Firstly with a statistical learning on a large number of samples, our approach can effectively address the difficulties caused by the changes of gesture and facial expression. Secondly the graph grammar rules are learned automatically from the sample data in a simple and natural way, which can be easily extended to other image and video applications.

We automatically extract 36 facial feature points (Figure 7) from a frontal face video by a graph grammar based semantic analysis. The automatic extraction consists of the following steps: (1) learning face grammar rules from samples of face images, (2) detecting faces in a frontal face video using the face grammar rules, and (3) extracting the facial feature points in the detected faces. We elaborate on the three steps as follows.

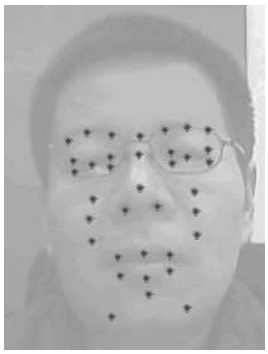


Figure 7 Facial feature points

A face grammar consists of face grammar rules, which can be expressed by the chiaroscuro patterns in human faces. See Figure 8 and Figure 9. The process to generate a face grammar rule from a face image is as follows. Firstly we compute the level set components of the image and select salient bright and dark components which satisfy required conditions on the colors, areas, intensity difference with the background, and shape regularity. The shape regularity is computed as

$$A_m / A_{c_2}. \tag{1}$$

where A_m is the area of the component, and A_{c_2} is area of its convex hull. Secondly we compute the weight centers of the selected salient components, as well as the Delaunay triangulation of these weight centers. For each edge in the triangulation, we find the relationship on the distance, direction, and area between

(a)

(b) (c)

Figure 8 (a) A face (b,c) Chiaroscuro patterns

the two salient components connected by the edge. Thirdly the grammar rule is generated so that it is expressed by the salient components and their mutual relationships. See Figure 9.

Figure 9 Facial feature points

The process to generate a group of face grammar rules from a set of face images is as follows. Firstly we encode each sample image by a sparse coding method [34], and then make compactness based clustering on the sample images [35]. Secondly we choose a representative image in each cluster. Thirdly we generate a face rule from each representative image.

After we have a group of face grammar rules, the process to detect faces in a frontal face video is as follows. Firstly we start with the first frame of the video to detect faces. Once a face is detected, we call the tracking procedure (to be introduced in the next section) to semantically track the detected face in the following frames. Secondly when detecting in a video frame, we use all face grammar rules and choose the best matched face. Thirdly when applying a grammar rule on a video frame, we find in the frame all qualified salient bright and dark level set components, and compute their weight centers, as well as the Delaunay triangulation of these weight centers. We then use the grammar rule to match the triangulation and find a matched sub-graph. In graph matching, we compare the vertices and edges of the graphs. The comparison of the vertices (salient components) is based on their area, shape, and color. The comparison of the edges is based on the distance, area ratio, and direction difference between the salient components connected by the edges.

Facial expression is mainly characterized by the positioning and deformation of the facial landmarks, e.g. eyes, nose, and mouth. Feature points around these organs are especially important for facial feature recognition. 20 to 30 feature points such located provide sufficient information for most tasks of facial feature analysis. Our algorithm extracts 36 feature points around the eyes, nose, and mouth in a detected face. The process is elaborated as follows. Firstly we manually locate the 36 points in each representative face image. Secondly for each feature point, we find 3 closest salient components in the grammar rule generated by the representative image. The weight centers of the 3 components are labeled as A,B, and C, respectively. We express the position of the feature point as

$$F = \alpha A + \beta B + (1 - \alpha - \beta)C. \quad (2)$$

Thirdly when we detect a face in a video frame by a grammar rule, the corresponding salient components will also be found. Fourthly for each feature point, in the video frame, we get the 3 corresponding salient components, whose weight centers are labeled as $\underline{A}$, $\underline{B}$, and $\underline{C}$, respectively. Then the location of the feature point in the video frame is computed as

$$\underline{F} = \alpha \underline{A} + \beta \underline{B} + (1 - \alpha - \beta)\underline{C}. \quad (3)$$

3. Semantics-based Object Tracking

Tracking an object in a video sequence is a process of continuously identifying the location of the object. Traditional tracking algorithms often fail due to accumulated errors caused by the continuous changes of the target object's shape, size, and moving direction.

To solve this problem, we propose a semantics-based object tracking approach, where the semantics is embodied in the fusion of low-level features using the

(a)

(b)

(c)

Figure 10 (a) A face image. (b) The light and dark stripes. (c) The key points.

graph grammar rules. Our algorithm consists of two phases: key point tracking [36] and tracking by graph grammar rules. We first use a level set method to compute the key points of the video frames and track the key points by a Kalman filter. Then the graph grammar rules are used to dynamically examine and adjust the tracking procedure to make it robust.

3.1 Key Point Tracking

We compute the key points in a video frame as follows. First we compute the level set components of the frame, and perform a region thinning on the components to get light and dark stripes (Figure 10b). Secondly we take the end points of the stripes as the key points (Figure 10c). All key points, as well as their colors and the tangent directions of the stripes at the points, are fed to a Kalman filter [37] for object tracking.

Kalman filter assumes Gaussian distribution of states and noise, and continuously uses imprecise data to update the best estimate of a linear or nearly linear system's current state. Let x be the state, ε the process noise, z the measurement, δ the measurement noise. Then we have:

$$x_t = A_t x_{t-1} + \varepsilon_t, \tag{4}$$

$$Z_t = C_t x_t + \delta_t. \tag{5}$$

The Kalman filter estimates the state x by (the values with bar on the top are predicted value and Σ the error covariance),

$$\bar{x_t} = A_t x_{t-1}, \tag{6}$$

$$\bar{\Sigma_t} = A_t \Sigma_{t-1} A_t^T + R_t. \tag{7}$$

and corrects the prediction by (K is Kalman gain)

$$K_t = \bar{\Sigma_t} C_t^T (C_t \bar{\Sigma_t} C_t^T + Q_t)^{-1}, \tag{8}$$

$$x_t = \bar{x_t} + K_t(z_t - C_t \bar{x_t}), \tag{9}$$

$$\Sigma_t = (I - K_t C_t)\bar{\Sigma_t}. \tag{10}$$

The tracking process is summarized in Figure 11.

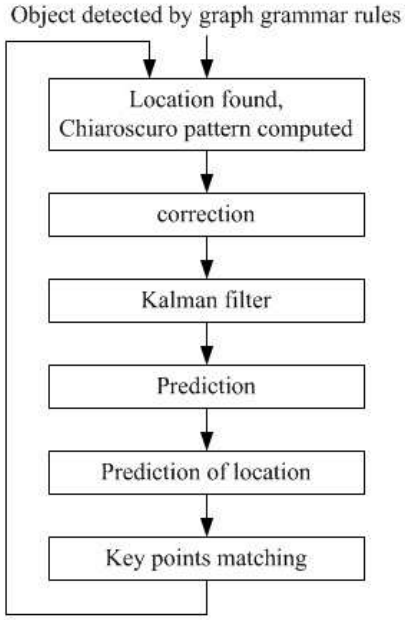


Figure 11 Flow chart of the key point tracking algorithm

3.2 Tracking By Graph Grammar Rules

Similar to other traditional tracking algorithms, the Kalman filter based key point tracking also has cumulative errors, which can cause drift away from the target object. To solve this problem, we introduce a tracking method based on graph grammar rules. In the object detection stage, we use graph grammar rules to find in the video frames the salient level set components which constitute the target object. In the object tracking stage, for each salient component we find in the following frames one or more corresponding components based on the results of the key point tracking. Then we select the best matched component by the grammar rules. Thus in the following frames we find the components which constitute the target object. If in one of the following frames we fail to find the object, or some key points miss, then we re-do the object

detection process in the frame to find the target object. In this way we use the graph grammar rules to dynamically examine and adjust the tracking procedure and make it robust. See Figure 12 for the flow chart.

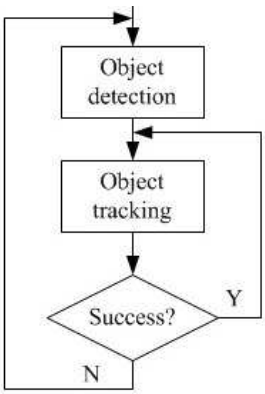


Figure 12 Flow chart of the key point tracking algorithm

For facial expression extraction, the overall velocity and trajectory of the human face are not important; what we really need is the relative motion among the facial feature points. Our graph grammar rules based tracking method can achieve robust tracking of the salient light and dark components, on the basis of which we can effectively track the facial feature points and compute their relative movement so that the afterwards facial expression extraction and analysis can be supported.

4. Experiments

We implemented our algorithm in VC++ and ran experiments on a Thinkpad X61 Notebook (Core 2 Duo T8100 @ 2.1GHz, 0.97GB RAM). For automatic facial feature points detecting and tracking, we tested with video data containing different facial expressions performed by 20 participants. Our algorithm automatically extracts and tracks 36 facial feature points. The overall accuracy is calculated as the ratio of the number of correctly detected/tracked feature points to the total number of feature points in all frames. In the experiments, our overall accuracy was 95%, and the tracking speed was 10 frames per second. The human visual system could process 10 to 12 separate images per second, perceiving them individually. So the speed of our algorithm met the need of facial tracking in real applications. See Figure 13 for sample frames with the facial feature points marked. For license plate recognition and tracking, we tested with 20-hour road surveillance video data. The overall accuracy of the system was 93%, and the tracking speed was 7 frames per second, which also met the need for tracking in real road surveillance videos. See Figure 14 for sample frames with the license plate marked.

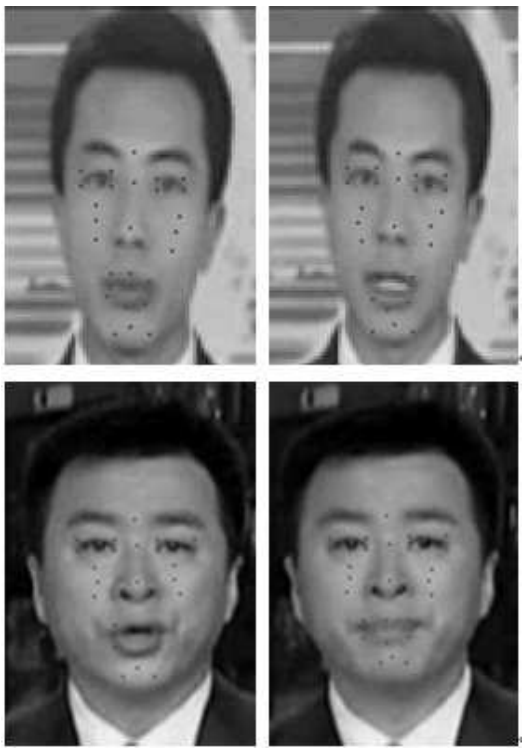


Figure 13 Sample frames with the marked facial feature points

5. Conclusion

Traditional tracking algorithms often fail due to accumulated errors caused by the continuous changes of the target object's shape, size, and moving direction. In object tracking, the target object is dynamic; its shape and characteristics change over time. The graph grammar model is more suitable than other formal tools to describe the dynamic characteristics of the target object.

In this paper we introduce a graph grammar based method to fuse the low level features and apply them to object detection and tracking. In our algorithm, the graph grammar rules are used to detect the object in the beginning of the video sequence and then dynamically examine and adjust the tracking procedure to make it robust. The effectiveness of the algorithm has been demonstrated by experiments.

Acknowledgment

This research was supported by Natural Science Foundation of China under contracts No. 61070112 and

Figure 14 Sample frames with the license plate marked

No. 61070116, and Hi-Tech Research and Development Program of China (863 Program) under contract No. 2009AA01Z317.

References

[1] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "An adaptive color-based particle filter", *Journal of Image and Vision Computing*, vol.21(1), pp.99-110, 2003.

[2] M. Isard and A. Blake, "Condensation-conditional density propagation for visual tracking", *International Journal of Computer Vision*, vol.29(1), pp.5-28, 1998.

[3] B. Ristic, S. Arulampalam, and N. Gordon,"Beyond the Kalman filter: particle filter for tracking applications", *Artech House Publishers*, 2004.

[4] I. Matthews, T. Ishikawa, and S. Baker, "The template update problem", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.26(6), pp.810-815, 2004.

[5] S. Avidan, "Support vector tracking", *Pattern Analysis and Machine Intelligence*, vol.26(8), pp.1064-1072, 2004.

[6] F. Senekal, "Fast region-based object detection and tracking using correlation of features", *21st Annual Symposium of the Pattern Recognition Association of South Africa (PRASA)*, Stellenbosch, South Africa, 2010.

[7] M. Yokoyama and T. Poggio, "A contour-based moving object detection and tracking", *Proceedings of Second Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (in conjunction with ICCV 2005)*, Beijing, China, October, 2005.

[8] D. Freedman and T. Zhang, "Active Contours for tracking distributions", *IEEE Transactions on Image Processing*, vol.13(4), 518-526, 2004.

[9] A. Yilmaz, X. Li, and M. Shah, "Contour Based Object Tracking with Occlusion Handling in Video Acquired Using Mobile Cameras", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.26(11), 1531-1536, 2004.

[10] A. Schindler, "Model-based detection and tracking of objects using a 3D-camera", *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2010.

[11] V. Kyrki and D. Kragic, " Integration of Model-based and Model-free Cues for Visual Object Tracking in 3D ", *IEEE International Conference on Robotics and Automation. ICRA'05*, Barcelona, Spain, 2005.

[12] A. Petrovskaya and S. Thrun, "Model based vehicle detection and tracking for autonomous urban driving", *Autonomous Robots*,vol.26(2-3), pp.123-139, 2009.

[13] S. Intille and A. Bobick, "Closed world tracking", *Proceedings of the 5th International Conference on Computer Vision. Boston, MA*, pp.672-678, 1995.

[14] P. Tissainayagama and D. Suter, " Object Tracking in Image Sequences Using Point Features", *Pattern Recognition*, vol.30, pp.105-113, 2005.

[15] A. Jepson, and M. Black, "Mixture models for optical flow computation", *Proceedings of IEEE Conference on Computer Vision Pattern Regnition*, New York, 1993.

[16] M. Nagl, "Formal languages of labeled graphs." *Computing*, vol.16, 1976.

[17] M. Nagl, "A tutorial and bibliographical survey on graph-grammars", *V. Claus, H. Ehrig, and G. Rozenberg, editors, Graph-Grammars and their Application to Computer Science and Biology*, pp.70-126, Springer-Verlag, Berlin, 1979.

[18] J. Egar and M. Musen, "Graph-Grammar Assistance for Automated Generation of Influence Diagrams", *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pp.235-241, 1993.

[19] R. Bardohl, M. Minas, A. Schurr, and G. Taentzer, "Application of graph transformation to visual languages",*Handbook of Graph Grammars and Computing by Graph Transformation, volume II: Applications, Languages and Tools*, World Scientific, pp.105-180, 1999.

[20] M. Nagl, A. Schrr, and M. Mnch (Eds.), "Applications of Graph Transformations with Industrial Relevance", *LNCS 1779 (ISBN 3-540-67658-9)*, Springer, Heidelberg, 2000.

[21] X. Han, X. Zeng, Y. Zou, and K. Zhang, "Survey of Graph Grammars", *Computer Science*, vol.35(8), 2008.

[22] J. Pfaltz and A. Rosenfeld, "Web Grammars", *Proceedings of First International Joint Conference on Artificial Intelligence*, 1969.

[23] Y. Song and D. Yue, "Multi-Feature Fusion for Video Object Tracking", *Proceedings of the 5th International Conference on Intelligent Networks and Intelligent Systems*, 2012.

[24] S. Tao, S. Wang, Y. Zheng, and Z. Huang, "CAD model retrieval based on inexact graph matching", *Journal of Computer-Aided Design & Computer Graphics*, vol.22(3), pp.545C552, 2010.

[25] A. Nagare, "License Plate Character Recognition System using Neural Network", *International Journal of Computer Applications*, vol.25(10), 2011.

[26] M. Khalil, "Car Plate Recognition Using the Template Matching Method", *International Journal of Computer Theory and Engineering*, vol.2(5), pp. 1793-8201, October 2010.

[27] M. Aksoy, G. Cagil, and A. Turker, "Number plate recognition using inductive learning", *Robotics and Autonomous Systems*, vol.33, pp.149-153, Canada, 2000.

[28] S. Sural and P. Das, "Fuzzy Hough transform and an MLP with fuzzy input/output for character recognition", *Fuzzy Sets and Systems*, pp.489-497, 1999.

[29] J. Xie, T. Liu , T. Wen, and W. Yan, "A Method for License Plate Recognition Based on Multiple Features and Weighted Pattern Similarity Measurement", *Computer Engineering & Science*, vol.30(8), pp.36-38, 2008.

[30] Y. Song and L. Shi, " License Plate Location by Level Set Transform and Voronoi Diagram", *Proceedings of the 4th International Conference on Intelligent Networks and Intelligent Systems*, Nov. 2011.

[31] Y. Song and S. Jin, " Matching Sequences of Salient Contour Points Characterized by Voronoi Region Features", *Visual Computer:International Journal of Computer Graphics*, vol.28(5), pp.475-491, 2012.

[32] Z. Zheng, J. Jiong, D. Chunjiang, X. Liu, and J. Yang, " Facial feature localization based on an improved active shape modal", *Information Seiences*, vol.178(9), pp.2215-2223, 2008.

[33] G. Edwards, C. Taylor, and T. Cootes, " Interpreting face images using active appearance models", *Proceeding of the International Conference on Face and Gesture Recognition*, pp.300-305, 1998.

[34] Z. Cui, S. Shan, X. Chen, and L. Zhang, " Sparsely Encoded Local Descriptor for Face Recognition", *International Conference on Automatic Face and Gesture Recognition*, 2011.

[35] Y. Song, S. Jin, " A Unique Property of Single-link Distance and Its Application in Data Clustering", *Data and Knowledge Engineering*, vol.70(11), pp.984-1003, Nov. 2011.

[36] Y. Song, " Object Tracking by Chiaroscuro Patterns", *Proceedings of the 2010 IEEE International Conference on Information Reuse and Integration*, 2010.

[37] R. Brown and P. Hwang, "Introduction to Random Signals & Applied Kalman Filtering, Second Edition", *John Wiley and Sons, Inc.*, 1992.