User Oriented Language Model for Face Detection

Daesik Jang
Dept. of Computer Information Engineering
Kunsan National University, Gunsan, South Korea
dsjang@kunsan.ac.kr

Gregor Miller, Sid Fels, and Steve Oldridge
Human Communication Technology Laboratory
University of British Columbia, Vancouver, Canada
http://hct.ece.ubc.ca/

Abstract

This paper provides a novel approach for a user oriented language model for face detection. Even though there are many open source or commercial libraries to solve the problem of face detection, they are still hard to use because they require specific knowledge on details of algorithmic techniques. This paper proposes a high-level language model for face detection with which users can develop systems easily and even without specific knowledge on face detection theories and algorithms. Important conditions are firstly considered to categorize the large problem space of face detection. The conditions identified here are then represented as expressions in terms of a language model so that developers can use them to express various problems. Once the conditions are expressed by users, the proposed associated interpreter interprets the conditions to find and organize the best algorithms to solve the represented problem with corresponding conditions. We show a proof-of-concept implementation and some test and analyze example problems to show the ease of use and usability.

1. Introduction

Face detection is one of the most important biometric problems in Computer Vision and has been researched for more than 20 years; as a result, the current status of face detection technology is well advanced. Many commercial applications of face detection are also available such as autofocusing for digital cameras, surveillance systems, image and film processing. Some algorithms also have shown impressive real-time performance and such recent advances of these algorithms have also made significant contributions in detecting other objects such as humans/pedestrians, and cars as well.

However, face detection is still challenging because it needs to cover all possible appearance variations caused by changes in illumination, facial features, occlusions, etc. In addition, it has to detect faces that appear at different scales, poses and in-plane rotations. These various conditions for face detection creates a large problem space for which many algorithms and implementations were published to solve different subspaces within this larger one. One problem encountered is that it is not easy to choose or combine optimal algorithms for each application. Even though there are many open source or commercial libraries to help developers solve their particular face-detection problem, they remain hard to use because they require specific knowledge of the details of the algorithmic techniques available.

The purpose of this paper is to come up with a high-level language model for face detection with which users can develop systems easily and even without specific knowledge of face detection theories and algorithms. By doing this, the problem of selecting algorithms and deciding complicated parameters for algorithms are isolated from development of face-detection applications. Developers just need to define the problem and express it with the language model suggested and an interpreter will select algorithms appropriate for the associated sub-space of the problem. We first consider the important conditions to categorize the large problem space of face detection. The conditions identified here are then expressed in terms of a language model so that developers can use them to express various requirements of a given problem. Once the conditions are expressed by developers, the interpreter plays an important role to interpret the conditions to find and organize the optimal algorithms to solve the represented problem.

The model is a part of the Open Vision Language (OpenVL), a vision language that allows programmers to

describe their vision problem in terms of what it is they want to do, instead of how they want it done. A proof-of-concept is implemented and some example problems are tested and analyzed to show the ease of use and usability.

Section 2 introduces some important related works. Section 3 explains about the conditions used in this paper to design a user oriented language model. Section 4 shows the proof-of-concept implementation and validation of the language model with some example cases. Section 5 concludes with future work.

2. Related Work

Some survey papers are providing versatile approaches for different algorithms used to detect faces in images. King surveyed some modern approaches for face detection including support vector machines, AdaBoost and neural networks [3, 13]. Yang et al. also proposed a novel categorization for face detection algorithms into four main groups such as knowledge-based methods, feature invariant approaches, template matching methods, and appearance-based methods [14].

Most recently, appearance based approaches showed good performance in terms of detection rate and speed. They carry out the task by extracting certain properties (e.g., local features or holistic intensity patterns) of a set of training images acquired at a fixed pose (e.g., upright frontal pose). Additional algorithms have been proposed to learn these generic templates (e.g., eigenface and statistical distribution) or discriminant classifiers (e.g., neural networks, Fisher linear discriminant, sparse network of Winnows, decision trees, Bayes classifiers, support vector machines, and AdaBoost) [9, 7, 5, 8, 11].

These categorizations seem very novel and meaningful to understand the problem of face detection, but most of them are based on a mathematical and theoretical point of view and appropriate for research of computer vision specialists. However, these taxonomy and analyses are not appropriate for considering a user oriented language model especially designed for general application developers who don't have specific knowledge about every vision algorithm. Furthermore, these do not allow for developers to easily change algorithm as their problem space changes.

Currently open source or commercial libraries based on various approaches are being used for practical systems. For example, OpenCV is one of the most popular open source libraries for computer vision developers. It provides various and general purpose functions for image processing and computer vision algorithms, and they are very efficient and useful even for practical real-time applications [1]. OMRON has succeeded in developing a mechanism called as OKAO Vision for quickly detecting a face by assessing varied brightness levels of different parts of the human face, thereby minimizing the volume of information required for detection [6]. In general, these libraries were developed for the users who want to add vision functionalities to their systems. Because they have focused on performance and practical usage, it might be effective to use one of them for certain tasks. However, they are mainly based on algorithmic skills for performance, and still require users to learn specific algorithm knowledge before using them leading to frustration trying to develop expert knowledge of different algorithms to figure out which one to use.

Condition	Description
Size of Face	Minimum and maximum size of face
Pose	Relative camera-face pose
In-place Rotation	Rotation of the image
Occlusion of Face	Partial occlusion by other objects
Facial Conditions	Facial expression, age and gender
Imaging Conditions	Color, Illumination and so on

Table 1. Conditions and descriptions

3. Conditions of Face Detection

We considered some important conditions based on diverse variations of face detection problem as the first step to make a language model for face detection because they can be identified easily from a user's point of view without specific knowledge on theoretical algorithms. These conditions provide important clues to model the requirements of the users and construct selection mechanisms for the interpretation of those requirements. Effectively, the underlying principle is that users know how to describe their vision problem but lack detailed knowledge on how to solve it.

As illustrated in section 1, different conditions make the problem space of face detection enormous. Yang et al. presented some important conditions as factors to consider for face detection in his survey even though they did not consider the factors to analyze and categorize diverse algorithms [14]. We used the conditions presented by Yang et al. and refined them by analyzing open source and commercial algorithms together in terms of language modeling. Table 1 shows the conditions considered in this paper based on the previous papers and practical usages and are described next.

3.1. Size of Face

Most algorithms are defining the size of face they can detect and it is an important factor to design and develop an algorithm. The size of face is affecting the algorithms in two ways. Firstly, the size should be considered in the beginning of algorithm design and different algorithms have different capabilities in terms of the minimum size they can detect. Especially, learning based algorithms have to be careful selecting a positive set of faces and the size is an

important criteria to construct it. Secondly, the face size determines the search space for detecting faces in an image. Thus, knowing this *a priori* can affect the accuracy and efficiency of algorithms and they have a kind of trade-off relationship. Practical libraries also define the minimum face size in their specification in accordance with the algorithms they are using.

In this way, the size of face is an important condition for the face detection problem and it can be an important criteria to select appropriate algorithms for a given problem. Further, specifying face size constraints is relatively straightforward for a user.

3.2. Pose of Face

The images of a face vary due to the relative camera-face pose (frontal, 45 degree, profile, upside down), and some facial features such as an eye or the nose may become partially or wholly occluded [3]. As a result, such variations of pose derive many differences in the appearance of faces projected into images.

To keep up with these variations, some algorithms use different features for different poses or they use different training sets in the case of learning based approaches. Thus, the variations of poses is another important condition to interpret user requirements. According to different poses, different algorithms can be chosen or different learning set can be selected as a parameter of a specific algorithm.

3.3. In-place Rotation of Image

The variations coming from in-place rotation looks very similar to those from different poses of faces in the sense that it also results in different angles of faces in images. However, in-place rotation can be distinguished from poses because it is normally derived from the rotation of the image itself instead of the rotation of faces in 3D. In terms of algorithms, the variations from in-place rotation can be solved by providing more search spaces for faces in an image. More rotated windows for a certain pose, for example, can be a simple and intuitive solution even though it requires more time for searching. Some algorithms provide rotation invariant methods as well [9].

3.4. Occlusion of Face

Faces may be partially occluded by other objects. In an image with a group of people, some faces may partially occlude other faces [12]. Some faces can also be occluded by sun glasses or masks. These kinds of variations basically should be considered from the designing stage of algorithm development and they affect critically in selecting a proper algorithm for given requirements.

3.5. Facial Conditions

Some internal variations such as facial expression, age and gender also can affect the appearance of faces, but these internal variations seem to be less influential to the detection process itself relative to other variations. Many face detection algorithms are showing robust results against these internal variations [2, 12].

However, these kinds of internal variations still have important meaning as conditions for expressing users' requirements. Some applications need to understand the expressions, age or gender of a face. These information may not be used for face detection but may be considered as additional information users may expect to extract from face detection. Some algorithms were developed to extract these additional information from faces and they are normally separated from face detection algorithms and conducted in sequence after face detection.

3.6. Imaging Conditions

When the image is formed, factors such as lighting (spectra, source distribution and intensity) and camera characteristics (sensor response, lenses) affect the appearance of a face [4]. Color and illumination are representative conditions from these kinds of diverse imaging conditions. These conditions are considered in selecting proper features for designing an algorithm and they also influence in the stage of preprocessing of face detection algorithms. Some algorithms are using color information as an important clue to detect face regions in an image and some are neglecting color features by using gray scale images for color invariant detection of faces [4, 13]. Most algorithms are preforming a preprocessing task to get a normalized and enhanced distribution of brightness to compensate for the variations stemming from different illumination conditions.

Often, it is not an easy task for users to identify and define exact imaging conditions such as color and illumination explicitly. Even with the same camera, the imaging conditions can vary with different environmental situations. As a result, we concluded that we need to provide an implicit facility to deal with these conditions either through automatic calculation or using meta-data when possible.

4. Implementation of Language Model

We have implemented a proof-of-concept implementation of our user oriented language model. The language model for face detection is instantiated as a part of the Open Vision Language(OpenVL) [10] motivated from the success of computer graphics frameworks such as OpenGL. OpenVL provides declarative semantics rather than a procedural language model. Programmers set the state or context of an OpenVL description specifying the variations of their data as well as any constraints on the solution and then al-

low the OpenVL implementation to run to solve the user's problem.

To specify the variations and constraints of various face detection problems, the conditions identified in section 3 are expressed according to their properties respectively. The expression of conditions is described as a context and it becomes an important interface with which users can express their requirements for an application. We also propose a simple and effective architecture for the language model and where the expressed conditions are processed and used to select and organize proper algorithms for the desired results.

4.1. Expression of Conditions

The language model provides an explicit and consistent method for users to specify the conditions introduced in section 3. We describe these below.

Face size is expressed as a proportion of the size of the image. For example, if the minimum pixels of width of a face is 24 and the width of the given image is 320 pixels then the minimum size of face to detect can be expressed as 0.075 instead of 24 pixels. We use this approach rather than an absolute pixel size since we want to ensure that the specification is independent of image resolution. By doing so, the code can be reused for different image resolutions more easily. In contraast, many algorithms and libraries use pixel size [13] making it more challenging for reuse

The pose of face is expressed together using the well known rotation angles of 3D objects; roll, pitch and yaw. (Note: any rotation representation can be used, including a full rotation matrix or quaternions). Our approach is in contrast to many face detection algorithms that express poses with less expressive terms such as frontal, profile and upright faces. However, these expressions cover only a small number of variations of poses and therefore they are not enough to express the condition of pose completely. Our representation can be partitioned into a smaller discrete set using predefined ranges for the sake of convenience. Thus, our more mathematical approach provides completeness and flexibility for pose specification.

In-place rotation typically comes from the rotation of camera along the optical axis or rotation of the image itself. This rotation can be expressed as the rotated angle of upright face in image plane.

The occlusion of face can be expressed using the proportion of occlusion to the area of a face intuitively. If 30% of a face is occluded by a mask, it can be expressed as 0.3.

Among facial conditions, age can be expressed using a range of ages. Another facial condition, gender, can be expressed with some representative labels such as "MALE", "FEMALE" and "BOTH".

Imaging conditions such as color and illumination of image are not easy for users to identify and express explicitly as explained in section 3 even though they are critical conditions in selecting proper algorithms. So, we applied some methods to determine the imaging conditions automatically instead of expressing them as explicit conditions in these examples. However, users have access to these parameters if they choose to set them explicity.

Table 2 summarizes the expression of conditions.

4.2. Architecture of The Language Model

Figure 1 shows the overview of the proof-of-concept architecture of the language model proposed in this paper. The expressed conditions are represented in the form of a context. The conditions are input by users based on their explicit requirements or generated automatically by *Implicit Condition Generator* as discussed above. The *Implicit Condition Generator* analyzes the input data and then determine corresponding implicit conditions if necessary. In this way, the expressed conditions constitute the context.

The context is then analyzed by the *Interpreter* to select and organize the optimal algorithm for a given problem. The *Interpreter* is composed of three sub steps of *Algorithm Selector*, *Preprocessor Organizer* and *Parameter Organizer* as illustrated in figure 1.

The *Algorithm Selector* selects the most appropriate algorithm according to the given context. We used a simple rule-based method for the selection mechanism. The rules are constructed based on the specification of each algorithm. As this is a proof-of-concept, we did not explore the range of viable *Algorithm Selector* approaches that can optimize for speed, accuracy and so on. For example, one approach could be to run all algorithms in parallel and select the best.

Most algorithms often have their own preprocessing steps for algorithm specific needs. Thus, we preprocess input images to be prepared optimally for the selected algorithm. The *Preprocessor Organizer* is responsible for determining these necessary preprocessing steps for the selected algorithm.

Some algorithms and libraries are very sensitive to input parameters. From different parameters, the algorithms can be adjusted for specific problem space and also optimize its performance in terms of speed and accuracy. However, these parameters are also very difficult to tune for these algorithms, because they are highly algorithm specific and are generally only understood with deep knowledge of the algorithm. The *Parameter Organizer* analyzes the context and tries to find the optimal parameter set for the selected algorithm.

Once the optimal algorithm is selected and organized by *Interpreter*, the *Executor* runs the algorithm and make the proper output for the application.

Condition	Representation	Example
Explicit Conditions		
Size of Face	Minimum and maximum sizes proportional to image width	sizeFrom : 0.075, sizeTo : 0.5
Pose	Angles of roll, pitch and yaw	yawFrom : -15 degree, yawTo : 15 degree
In-place Rotation	Angles of rotated image	inplaceFrom : -45 degree, inplaceTo : 45 degree
Occlusion of Face	Proportion of occluded area in face	occlusion : 0.2 (20%)
Facial Conditions	Labels for age and gender	gender : FEMALE, age : YOUNG
Implicit Conditions		
Imaging Conditions	Illumination and Color	illumination : 10 lux

Table 2. Expression of conditions.

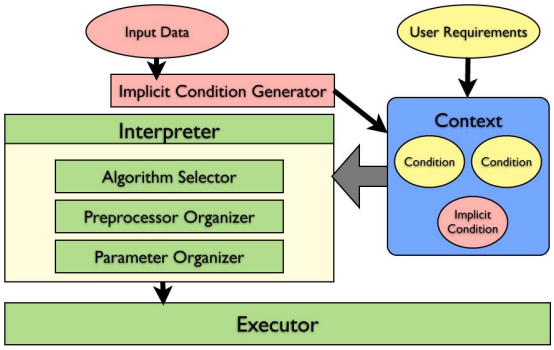

Figure 1. Architecture of the language model

conditions	rules(*if*)
Size	0.2 ~ 0.95
Pose(Roll)	-20 ~ 20
Pose(Pitch)	-45~ 45
Pose(Yaw)	-45 ~ 45
In-place Rotation	-20 ~ 20
Occlusion	0.3
then: *AdaBoost based*	

Table 3. Simple rules for algorithm selection

4.3. Validation

We present two different detection problems to validate and demonstrate the ease of use of our proof-of-concept language model proposed in this paper. Three different face detection algorithms were implemented for the selection of proper algorithms in this paper: AdaBoost based algorithm [13], Neural Network based algorithm [9], and Color based algorithm [4]. Some parts of the simple *"if-then"* style rules used for algorithm selection for the proof-of-concept implementation are illustrated in table 3.

The first case is to detect an upright, frontal and large face for face identification. Face detection is often used as a preprocessing for identifying persons by providing the exact area of face to recognize. In this case, most cameras

Conditions	Value
Size of Face	0.4 ~ 0.8
Pose(Roll)	-15 ~ 15 degree
Pose(Pitch)	-15 ~ 15 degree
Pose(Yaw)	-15 ~ 15 degree
In-place Rotation	-10 ~ 10 degree
Occlusion of Face	0.2
Facial Conditions	gender : BOTH, age : ALLAGE

Table 4. Example condition representation for face identification problem.

are installed in front of a door and the person to be identified is required to stand at a specific position so the face of the person should be acquired properly. As a result, the face in the image taken is well positioned and the size is big enough not only for detection but for recognition. Table 4 shows the conditions expressed to describe this problem. Figure 2 shows an example of C++ code for defining the problem of face identification problem using the proposed language model. Figure 3 shows the result of the execution of the code in Figure 2 through the proof-of-concept architecture of the language model and the AdaBoost based algorithm [13] was selected by *Algorithm Selector*.

The second case simulates face detection used for a surveillance system. Detecting faces and analyzing the activity is one of the important functions for intelligent surveillance. In this problem, faces are relatively far from the camera and the pose and angle of the face can not be guaranteed to be at a certain range. In that sense, the face detection should deal with small faces with arbitrary pose and in-place rotation. Table 5 shows the conditions expressed to describe this problem. As a result of the algorithm selection, Neural Network based algorithm [9] was selected for the second example.

5. Conclusion and Future Work

A novel approach for a user oriented language model for face detection is proposed in this paper. Important con-

```
FaceDetector detector;
FaceContext context;

context.sizeFrom = 0.4;
context.sizeTo = 0.8;
context.angleRollFrom = -15;
context.angleRollTo = 15;
context.anglePitchFrom = -15;
context.anglePitchTo = 15;
context.angleYawFrom = -15;
context.angleYawTo = 15;
context.inplaceFrom = -10;
context.inplaceFrom = 10;
context.occulusion = 0.2;
context.gender = BOTH;
context.age = ALLAGE;

detector.setContext(&context);
detector.Run(image, &context);

Segment face_region = context.segment;
```

Figure 2. Example of C++ code for face identification problem using the proposed language model

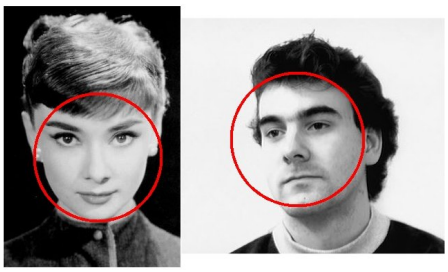

Figure 3. Result of face detection for face identification problem

Conditions	Value
Size of Face	0.075 ~ 0.2
Pose(Roll)	-45 ~ 45 degree
Pose(Pitch)	-45 ~ 45 degree
Pose(Yaw)	-90 ~ 90 degree
In-place Rotation	-90 ~ 90 degree
Occlusion of Face	0.2
Facial Conditions	gender : BOTH, age : ALLAGE

Table 5. Example condition representation for surveillance system.

ditions for the face detection problem that can be identified easily by users are investigated and the architecture for the language model based on these conditions were developed. The problem of selecting algorithms and deciding complicated parameters for algorithms are isolated from development with the proposed language model. Two example problems were expressed and tested using the proof-of-concept language model and demonstrated the ease of use and usability.

In the future, more face detection algorithms will be analyzed and added for more practical and richer usability of the language model. Some intelligent approaches for selecting algorithms is necessary to be considered for more optimal selection process.

References

[1] G. Bradski and A. Kaehler. Learning OpenCV: Computer vision with the OpenCV library, 2008. 2

[2] M. Castrillon, O. Deniz, D. Hernandez, and A. Dominguez. Identity and gender recognition using the encara real-time face detector. In *Conferencia de la Asociacin Espaola para la Inteligencia Artificial*, 2003. 3

[3] A. King. A survey of methods for face detection, 2003. 2, 3

[4] J. G. R. Maia, F. d. C. Gomes, and O. d. Souza. Automatic eye localization in color images. In *Proceedings of the XX Brazilian Symposium on Computer Graphics and Image Processing*, pages 195–204, 2007. 3, 5

[5] T. Matthew and A. Tal. Detecting, localizing and classifying visual traits from arbitrary viewpoints using probabilistic local feature modeling. In *Proceedings of the 3rd International Conference on Analysis and Modeling of Faces and Gestures*, pages 154–167, 2007. 2

[6] OMRON OKAO Vision. http://www.omron.com/. 2

[7] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997. 2

[8] S. Phimoltares, C. Lursinsap, and K. Chamnongthai. Face detection and facial feature localization without considering the appearance of image context. *Image and Vision Computing*, 25(5):741–753, 2007. 2

[9] H. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, page 38, 1998. 2, 3, 5

[10] C. Shen and S. S. Fels. OpenVL: Towards a novel software architecture for computer vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007. 3

[11] C. Tao, D. Shanxua, L. Fangrui, and R. Ting. Face and facial feature localization based on color segmentation and symmetry transform. In *Proceedings of International Conference on Multimedia Information Networking and Security*, pages 185–189, 2009. 2

[12] M. Toews and T. Arbel. Detection, localization and sex classification of faces from arbitrary viewpoints and under occlusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:1567–1581, 2009. 3

[13] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, page 511, 2001. 2, 3, 4, 5

[14] M. H. Yang and D. J. Kriegman. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002. 2