THE DIGITAL IMAGE EXPERIMENT PLATFORM BASED ON EVEN LINER GRAMMAR

Limin Ao [1,2] Bo Li [2] Yongsheng Chen [2]

1. College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, China

2. Department of Information Engineering, Northeast China Institute of Electric Power Engineering, Jilin 132012, China

Abstract: This paper describes a digital image processing experiment platform which is on the theory basis of the Even Linear Grammar. The platform is based on the Automatization of COM, which can increase the reusability of the image processing methods. In the image processing process, an Image Processing Process Tree (IPPT) is generated, which can make the image analysis easy.

Key words: syntax-directed, COM, IPPT, Reusable, Dynamically Construct

1. INTRODUCTION

In the process of image processing, in order to achieve our goal, we have to do many experiments and research, even to modify or design arithmetic to meet our need. In existing image processing software, Photoshop can't modify or add arithmetic; Matlab can do this, but everything must be preestablished, it is impossible to add some arithmetic into a program dynamically, and it can't record the middle results.

This paper is trying to describe an image processing platform which is based on constructed language and Syntax-directed Method.

Even Liner Grammar [1] is used to describe a constructed language. We can program using constructed language.

In general, an image-processing program is composed of some common image-processing methods. We can make use of the structure of attribute

grammar to reconstruct the image processing methods to be semantic functions of attribute grammar.

One of the ways to increase the reusability of the image processing method is to use COM.

COM [2] is short for Component Object Model. COM allows the COM components communicate alternately in a uniform way.

2. EVEN LINER GRAMMAR AND SYNTAX-DIRECTED METHOD

Let F be an alphabet. A ELG(Even Liner Grammar) whose form is G=(N,Σ,P,X) is a context-free grammar. Its productions have the form like:

$$A \to uBv \quad \text{or} \quad A \to w$$

Thereinto, A, B ∈ N; u, v, w ∈ Σ*; |u| = |v| . Grammar G is a linear grammar. N is a non-terminal symbols set, Σ is the terminal symbols set, P is the set of productions and X is the started non-terminal symbol.

The definition of the language based on Grammar G:

$$L = L(G) = \{s \mid s \in \Sigma^*, X \overset{*}{\Rightarrow} s\}$$

Thereinto, the form $X \overset{*}{\Rightarrow} s$ expresses that the string s can be inferred from the started non-terminal symbol X by a serious of productions.

Takada expresses his viewpoint in literature [3]: the inference of ELG can be come down to the inference of DFA(Deterministic Finite Automata).

Takada's thinking:

There is an alphabet F and a ELG G=(N,Σ,P,X). Sign every production in P with a single letter in F. Let $f=f_1f_2...f_k \in F^*$. the form $X \Rightarrow f_G\omega$ expresses that the string ω can be inferred by the productions $f_1f_2...f_k$ from the started non-terminal symbol X of G .Let C be a language on the alphabet F, then the language generated by the Grammar G on the control set C can be defined as:

$$L_C(G) = \{\omega \in \Sigma^* \mid X \overset{*}{\Rightarrow} f_G\omega, f \in C\}$$

So every sentence in the control set C corresponds to the inference process of a sentence in $L_C(G)$.

Takada proves that:

Any Even Liner Language L can be generated by a universal ELG G^0 on a control set C which is a regular language. L and G^0 are all on an alphabet Σ. That is to say, for any ELL(Even Liner Language) L, there is a form like:

$$L = L_C(G^0) = \{\omega \in \Sigma^* \mid X^0 \overset{*}{\Rightarrow} f_{G^0}\omega, f \in C\}$$

Thereinto, C is a regular grammar, G^0 is defined as

$$G^0 = (\{X^0\}, \Sigma, P^0, X^0).$$

$P^0=\{X^0\rightarrow aX^0b|a,b\in\Sigma\}\cup\{X^0\rightarrow ab|a,b\in\Sigma\}\cup\{X^0\rightarrow a|a\in\Sigma\}\cup\{X^0\rightarrow\varepsilon\}$[1].

Takada's thinking is used in this experiment platform.

Syntax-directed Method is the typical method to analyze the meaning of a sentence. The Syntax-directed Method is used in this experiment platform alternately in the image processing process. The platform uses the Syntax-directed Method to determine the style of the operation, so as to construct an Image Processing Process Tree (IPPT) dynamically. A node in IPPT is an image. If there is a path between two nodes, it means that one image is acquired form the other image by some image processing method. The methods used in the IPPT can form a program. So the program represents the image processing process.

3. BRIEF INTRODUCTION OF COM

What COM (Component Object Model) provides is not only the criterion for components alternation, but also the alternation environment.

a) Specialities of COM

Theoretically speaking, COM is a criterion about C/S.

COM has the specialities of Language Independence, Robust Versioning, Local Transparency and Object Orientation.

b) Automatization

Automatization is a special example of COM. It is based on COM, but its use is much wider than COM. Automatization provides normal attempering measures for the transmission of parameters and return values between different processors or different computers.

c) Later Binding

Later Binding is one of the important characteristics of Automatization. That is, the client program doesn't check the data type of the parameters when it is programmed, but check when the component is used.

4. THE THEORY BASIS OF THE IMAGE PROCESSING PROCESS

In the experiment platform, along with the image processing process, a image processing process tree (IPPT) can be constructed. The root of the tree is the original image, the middle nodes are the images generated in the processing process, and the branches are the processing functions used in the platform. Users of the platform can easily know how an image is generated in the platform by along the branches of IPPT.

Let R_1 be the original image, R_2, R_3,..., R_n be the generated images in the image processing process, f_1, f_2, ..., f_n be some function in the processing process. Then, there is a IPPT:

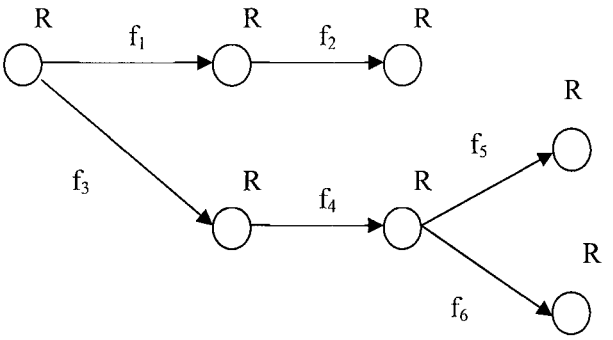

Figure 1. An IPPT

As figure1 shows, the original image R_1 can generate a serious of images by the image processing process. Supposing R_7 is the image that the user wants. The user can know how R_7 is acquired by analyzing the IPPT. And the user can save the process of generating R_7 as one operation, so as to simplify the operation process when it is used next time.

At the same time, every image generated in the processing process can be used as a root node to generate another IPPT, and the tree can go on and on.

Formalize the image processing process:

Another image will be generated if an image is processed by a specific arithmetic. And the generated image can be processed again. Let X,X',X'' represent an image, and a,b,...represent some specific image processing arithmetic, then the image processing process can be expressed as:

$$X \rightarrow aX', \quad X' \rightarrow bX''.$$

To the image experiment platform, image is just data, and an image processing arithmetic is just an operation.

So the image processing process can be described as: $X \rightarrow \alpha X$.

Thereinto, X is some image, α is some image processing arithmetic.

So every operation in the experiment platform can be considered as an application of regular grammar. RLG is a kind of regular grammar. Let a operation in the platform be f_i.

Thereout, we can define a regular language C, its corresponding regular grammar is: $G_C = (N_C, \Sigma_C, P_C, X_C)$. Thereinto, $N_C = \{X_C\}$, $\Sigma_C = \{ f_1, f_2, ..., f_k \}$, $P_C = \{ X_C \rightarrow f_i X_C \mid f \in \Sigma_C \} \cup \{ X_C \rightarrow f_i \mid f_i \in \Sigma_C \}$

The image experiment platform provides the function to save a serious of image processing operations to be one operation. That is to say, define a new operation f, let $f = f_1 f_2 ... f_k$. if F^* is the set of all the combinations of the

arithmetic in the platform, then $f \in F^*$. And f can be inferred in regular language C, so $f \in C$.

On the other hand, the operation result can be described in 3 ways: the image data will be written into database, the DDB data to display the image in the screen, and the record about the current operation generated in the image processing process. So an operation f_i can be described as $X \rightarrow \alpha X' \beta$, thereinto, X is the original image, X' is the generated image, α is the image data written into the database, and β is the operation record. $|\alpha|=|\beta|=1$. When we get the image we want, it's not necessary to process any more. We just need to save the image data into database and record the information about the current arithmetic. In that case, the operation can be described as

$$X \rightarrow \alpha \beta \text{ or } X \rightarrow \alpha.$$

Thereout, we can get an ELG G^0, which is suited to this image processing experiment platform:

$$G^0 = (\{X^0\}, \Sigma, P^0, X^0)$$

Thereinto, $P^0=\{X^0 \rightarrow \alpha X^0 \beta | \alpha, \beta \in \Sigma\} \cup \{X^0 \rightarrow \alpha\beta | \alpha, \beta \in \Sigma\} \cup \{X^0 \rightarrow \alpha | \alpha \in \Sigma\}$.

According to Takada's proof, ELG G^0 can generate ELL $L_C(G^0)$ on the control set C.

$$L_C(G^0) = \{\omega \in \Sigma^* \mid X^0 \overset{*}{\Rightarrow} f_{G^0}\omega, f \in C\}$$

ω is a combination of the operation results.

Analyze the IPPT in figure1:

The production set of ELG G^0 is:

f_1: $R \rightarrow \alpha_1 R \beta_1$

f_2: $R \rightarrow \alpha_2 R \beta_2$

f_3: $R \rightarrow \alpha_3 R \beta_3$

f_4: $R \rightarrow \alpha_4 R \beta_4$

f_5: $R \rightarrow \alpha_5 R \beta_5$

...

f_n: $R \rightarrow \alpha_n R \beta_n$

If the number of image processing functions in the platform is n, the number of production as above is n. In the production, the non-terminal symbol is X. At the same time, X is the started non-terminal symbol.

α_i and β_i are the terminal symbols in the productions. α_i represents the image data written into database, β_i represents the current operation record.

We plight here, when it is not necessary for an image to be processed more, we use the production $X \rightarrow \alpha$ to acquire the image data.

So, we need to define a production f_0: $R \rightarrow \alpha$.

So in figure 1, different path represents different processing process.

The inference process of R_3 :

$$R \overset{f_1}{\Rightarrow} \alpha_1 R \beta_1 \overset{f_2}{\Rightarrow} \alpha_1 \alpha_2 R \beta_2 \beta_1 \overset{f_0}{\Rightarrow} \alpha_1 \alpha_2 \alpha \beta_2 \beta_1$$

The inference process of R_6 :

$$R \xRightarrow{f_3} \alpha_3 R\beta_3 \xRightarrow{f_4} \alpha_3\alpha_4 R\beta_4\beta_3 \xRightarrow{f_5} \alpha_3\alpha_4\alpha_5 R\beta_5\beta_4\beta_3 \xRightarrow{f_0} \alpha_3\alpha_4\alpha_5\alpha\beta_5\beta_4\beta_3$$

The inference process of R_7 :

$$R \xRightarrow{f_3} \alpha_3 R\beta_3 \xRightarrow{f_4} \alpha_3\alpha_4 R\beta_4\beta_3 \xRightarrow{f_6} \alpha_3\alpha_4\alpha_6 R\beta_6\beta_4\beta_3 \xRightarrow{f_0} \alpha_3\alpha_4\alpha_6\alpha\beta_6\beta_4\beta_3$$

Supposing that R_7 is the image we want. We can know that R_7 is acquired from R_1 via the image processing arithmetic f_3, f_4, f_6. Let $f_G^0 = f_3 f_4 f_6$, then the inference process of R_7 can be described as:

$$R_1 \overset{*}{\Rightarrow} f_{G^0} R_7$$

f_G^0 can be inferred by the regular Grammar C, so $f_G^0 \in C$.

So it is can be considered that the inference process of R_7 is a sentence in $L_C(G^0)$ which is generated by ELG G^0 on the control set C.

So we can say that the image experiment platform is an application of ELL $L_C(G^0)$.

5. THE IMPLEMENT OF THE PLATFORM

5.1 System Framework of the Platform

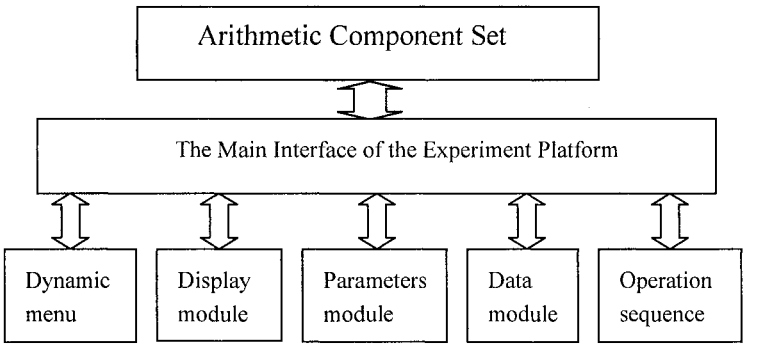

Figure 2. System Framework

The system framework is showed in figure 2. The ability of every component is:

(1) Arithmetic component set (P^0): the set of the arithmetic about image processing including the arithmetic you programmed yourself.

(2) The dynamic menu module: implement the function to generate dynamic menus from the methods in COM components.

(3) Display module: to display the image in a new window, this is generated in the image processing process.

(4) Parameters acquired module: to acquire the information about parameters from the database, and generate a parameter dialog dynamically which is used to implement the alternation between the user and the program.

(5) Data-saving and data-query module: to save the data acquired from the experiment process, so as to the data can be query later. In fact, the data saved is the IPPT generated in the image processing process.

(6) Save and execute the operation sequence: this module can save a serial of operations to be one operation, and then execute it.

That is equal to define a new operation f, and let $f = f_1, f_2,\ldots f_n$, f_i is an operation.

5.2 The design and implement of the interface of COM server

In order to meet the demand of the platform, the design of the interface of the COM server must fit the condition below:

(1) The odd interface methods must have the same parameter form, and so as the even interface methods. By this way, the platform can process the method in two common ways.

(2) Every arithmetic in this platform is implemented in two interface methods. The odd method is used to retrieve the string which describes the function of arithmetic, and the even method implement the function of the arithmetic.

For example:

interface ILinear : IDispatch

{ [id(1), helpstring("gray linear truncation")] HRESULT GetHelpString_1([out,retval]VARIANT *lResult);

[id(2), helpstring("gray linear truncation ")] HRESULT GrayLinearOff([in]unsigned char *ArrayGray, [in]long nWidth, [in]long nHeight, [in]float *pfset, [out,retval]VARIANT *lResult);

..... }

The methods id(1) and id(2) complete the "gray linear truncation" function together. The method id(2) implement the arithmetic, and the method id(1) retrieves the string "gray linear truncation" which will be used as the name of a menu item in the platform.

5.3 The Retrospect Ability of the Platform

The processing object of every operation in the platform is the image displayed in a child window. And the image generated by the operation will

be displayed in a new child window. So the contrast between the two images will be much more pronounced, the original image can be reprocessed again.

In this way, every node in the IPPT generated in the image processing process can be processed as a root node.

5.4 Save the Function Sequence

Implement the ability like: define a new operation f, let f= f_1, f_2,... f_n.

The platform implements the ability by record the information about every operation.

For example:

In order to find the boundary of a target area, we should operate like this:

① gray linear truncation, generating image 1, record as "Linear_Off 100 255 0 255";

② finding the edge of image 1, generating image 2, record as "Sobel";

③ turning image 2 to white-black image :image 3, the threshold is 128. record as "Two_Value 128";

④ edge trace image 3, record as "Edge_Trace" .

Save the operations above in a file named "Get_object_edge.serial".

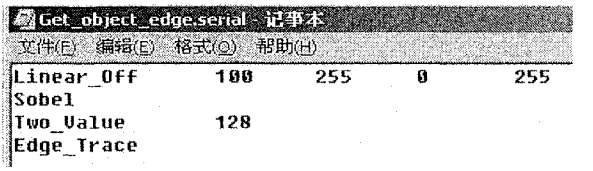

Figure 3. Get_object_edge.serial

If it is needed to perform the operation above again, the user just need to execute the Get_object_edge.serial. And the content in the file can be modified to be the appropriate content. This ability makes the operation more agile.

6. CONLUSION

The Image Processing Experiment Platform has powerful expansibility; it provides a reliable experiment platform for image processing.

REFERENCE

[1] ZHANG Rui-Ling. Grammatical Inference: Retrospect and Prospect. Journal of software. 1999, 10(8): 850 ~ 860

[2] Dale Rogerson. COM Technology Inside. Beijing: Publishing House of Tsinghua University, 1999

[3] Takada Y. Grammatical inference for even linear languages based on control sets. Information Processing Letters, 1988,28(4):193～199