A Pattern Language for semi-automatic generation of Digital Animation through hand-drawn Storyboards

Pedro H. C. Braga, Ismar F. Silveira
Programa de Pós Graduaçáo em Engenharia Elétrica e Computaçáo
Universidade Presbiteriana Mackenzie
São Paulo, SP, Brazil

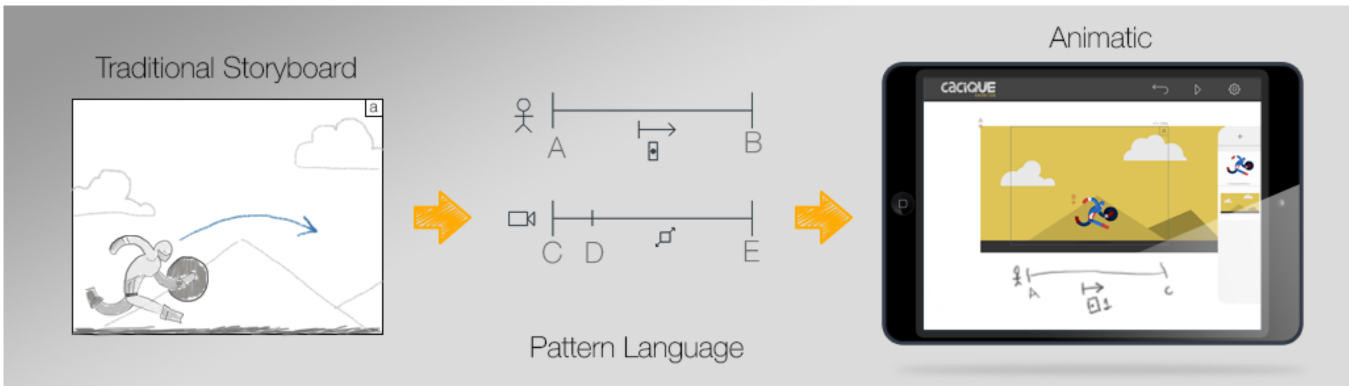


Fig. 1. Teasing result of our method: in a traditional storyboard frame (left), we add some annotations formated by our iconographic pattern language (middle) to generate a semi-automatic animation - animatic (right)

***Abstract*—The pre-production stage of the film is important to establish communication between directors and artists. One of the main documents generated in this step is the storyboard. This paper presents the proposal of a pattern language for storyboards, which aims to represent characters movements, cameras and elements of scene, enabling the generation of a semiautomated digital animation.**

***Keywords*-Computer Vision; Visual Language; Semiautomated Animation; Storyboard; Animation;**

I. Introduction

Storyboard is the tool that presents the context of a narrative by images, symbols and words that represent actions, intentions and emotions. “The key aspect of the visualisation process is storyboarding. Though this is intrinsically related to the script and soundtrack – considered later in this production section – it is addressed here as a logical continuity of the drawing and designing process. It is in the storyboard stage that visualisation is intrinsically linked to narrative – literally telling the story in pictures.” [1]

“Making films whether they are animated movies or real images requires careful organization of all the elements that make up the scene.” [2] No matter how many elements are in scene, a good composition of a storyboard helps to follow the plot and make the film better. It is probably the simplest way by which directors could communicate their ideas to the crew.

Storyboard creation is a unique process of a studio or an animators team, however that the symbolic language presented is restricted to a very specific domain and requires close attention when is translated to another team.

Contributions: The main goal of this work is to introduce a new approach on creating and understanding storyboards, by presenting an original visual language that can reproduce the major actions, movements and significant elements of a scene composition. Such language is meant to be interpreted in order to allow the generation of a semiautomatic animation using computer vision algorithms.

II. A Systematic SotA Review

The State of the Art (SotA) in the field of semiotics that focus on the practice of storyboarding or the definition of iconographic visual languages is presented below. This research seeks to know the main terminology adopted during this procedure and what are the main adopted symbols and their related meanings. A search for related works was performed on the following databases: SCOPUS, ACM-IEEE and Science Direct. This methodological selection is justified by the fact that they conform the most relevant sources in this knowledge field for scientific articles published in indexed journals. After all SotA-related systematic review procedures, only two works were proven to be strictly related to the present proposal:

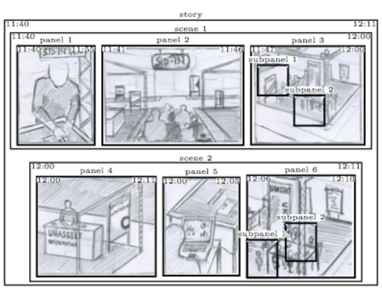
Timisto [3] and DirectorNotation [4]. Both will be briefly commented in the following sub-itens:

A. Timisto: a technique to extract usage sequences from storyboards [3]

Timisto presents a new approach to extracting relevant information in storyboards. The software developed, Time In Storyboard – TIMISTO – does not intend to interfere in the creative process, but maximize structured information on time, which can be inferred from a storyboard.

The work approach is based on work related to comic books, due to the great similarity between the two forms of publication. Both are compared for the provision of management and ease of understanding of the content through the way they are presented.

TIMISTO system does not interfere in creating a storyboard, but allows its creator to add time annotations in each frame. The author opted to use absolute time annotations, aiming to ease the understanding and avoid misunderstandings among the professionals. Fig. 2 shows how TIMISTO uses annotations on storyboards.



(a) Temporal notes on storyboards using TIMISTO formatting.

Fig. 2. Related Work - TIMISTO

B. DirectorNotation - Artistic and Technological System for Professional Film Directing [4]

Thinking about establishing a common artistic language that can be interpreted by a software feature, the authors established an artistic language supported by technical tools for planning and analysis.

Annotations are not merely a graphical interface, and tools do not have the intention substitute the directors decisions. The article presents the cultural motivations and current practices to justify the creation of such tools.

The basis for the symbols adopted by the language is Labanotation, a specific language for choreographic representation. It is a powerful language and widely used in arts.

The article also features an extensive background of possible uses of the language, including the creation of storyboards.

III. Pattern Language Fundamentals

Trying to extract the best of each related work and improve their weak points, a new language was created based on the following principles:

TABLE I
ONTOLOGY OF SYMBOLS

Group	Elements
Element	Scenario Character Camera light
Action	Swipe Dialog
Movement	Translation Rotation
Transition	Frame Translation Scene Translation
Shot	WiS – Wide Shot WS – Weather Shot VWS – Very Wide Shot TS – Two Shot POV – Point-of-View Shot OSS – Over the Shoulder Shot NS – Noddy Shot MS – Mid Shot MCU – Medium Close Up EWS – Extreme Wide Shot ECU – Extreme Close Up CU – Close Up CI – Cut In CA – Cut Away
Modifiers and Filters	Blur Shadow Glow Bevel and Emboss Alpha Color Transform

- The language must be natural for the storyboard artist;
- The iconography must be simple to draw and familiar to remember;
- Each symbol has a computational command associated to it;
- Items can be combined in order to produce a better result;
- A logical profile must be created, saved and reproduced by any software

An ontology of the problem domain was created to represent the language basis and it can be found at http://goo.gl/jy2C7e. The elements are thus cathegorized in six main groups, as seen on Table I.

“An image generally has three functions:

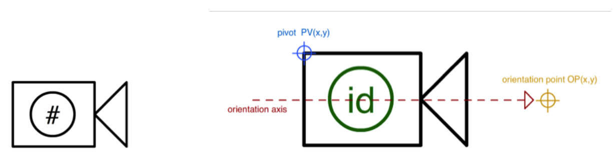
- a representative function, in which it makes present what is absent (the represented object) and the displayers showing off itself;
- a function of processing in which the image can act in the subject-spectator behavior, mobilize and energize their actions and representations: pictograms, illusory images, educational or propagandistic images;
- a climate function, in which contains the represented object and its viewer in the same housing, giving the viewer an illusion of a shared perception.” [5]

Thereafter, the language symbols must not only represent the ontology elements, but they need to have a specific behavior to engage the spectator in the story.

Semiotic principles, such as Gestalt laws, were observed to establish the best symbol for each element. The main idea is to “keep it simple and remarkable”.

IV. Dictionary Example

Each element of the computational language has a related symbol. A dictionary was created to produce the language basis. The symbol of a camera is presented bellow:



(a) Icon (b) Visual Representation

Fig. 3. Camera Symbol

There is a documentation for each language symbol, showing its icon – as in Fig. 3(a) –, name, code formatting, meaning, visual representation – as in Fig. 3(b) – the computational representation, usage, restrictions, obligatory combinations, optional combinations, symbolic references and theoretical references.

V. Main Grammar Points

The language is based on animation principles cited by Williams [6]. The main control of an attribute is a timeline, represented by an horizontal line, that may have as many subdivisions as many control frames the attribute requires. For example, the camera number one at Fig. 4 starts at point A and moves to frame A – a zoom in movement, doing a close up shot –, then it goes to frame B – a zoom out movement, generating a mid shot.

All attributes must have a timeline to indicate their properties on scene time. The annotations can be placed anywhere outside the frame. Elements like points, subframes and movement paths can be positioned on the frame. According to the ontology created, we established the minimum language symbols representing scenario, camera, character and light. Each of them must be chosen at the beginning of a timeline. This can be subdivided to represent the other groups of elements (action, movement, transition shots).

To add an effect to a section of the timeline, one should draw a modifier with the desired patch. The modified symbol can be positioned above or below the timeline, between the divisions that represent the start and the end points of the range.

Traditional animations can be settled outside the frame, using the animation symbol and drawing its frames side by side. So it could be used as a parameter for some character’s timeline, for instance. At Fig. 4, the blue timeline is for a character animation and it uses a path – drawn on the main frame. The character starts on the A side of the path and goes to the B side through the path. At the same time, it executes the frame-by-frame animation settled. The yellow timeline represents the scene lightning, that starts turned off, passes by half light before the half time and ends turned on.

It is important noticing that the timeline width is not important, but the spaces between its subdivisions are. Thus, the time is normalized and can be configured using the software configuration interface.

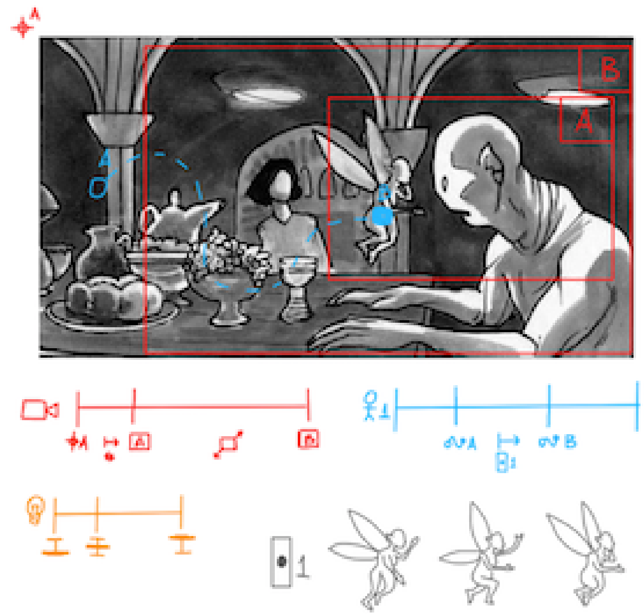


Fig. 4. Language Usage

VI. Computer Language

XML language was chosen to represent models in order to create an universal standard that could be read by different applications. In order to do it, a XML Scheme that depicts tags that are used and defined by the dictionary was created. A valid and well-formed document must follow the instructions of the Scheme and the descriptive grammar of the language.

The syntax of the language, therefore, should follow the foundations of the Standard Generalized Markup Language (SGML) to be Drawn up a well-formatted and valid document under the Scheme.

Concerning the example of a camera element, the recognition of the symbol builds the computational profile presented in Fig. 5.

```
<storyboard name='storyboard1'>
  <properties>
     <author>John</author>
     <studio>Studio X</studio>
     <framerate>24</framerate>
     <aspectratio>16:9</aspectratio>
  </properties>
  <frames>
     <frame name='a' duration='10'>
        <objects>
           <point name='A'>
              <position x='0' y='0'/>
```

```
            </point>
            <point name='B'>
               <position x='100' y='200'/>
            </point>
            <point name='C'>
               <position x='100' y='100'/>
            </point>
            <camera code='cam[1]'>
               <pivot>A</pivot>
               <orientation>C</orientation>
            </camera>
         </objects>

         <movements>
            <movement>
               <object>cam[1]</object>
               <startpoint>A</startpoint>
               <endpoint>B</endpoint>
            </movement>
         </movements>
      </frame>
   </frames>
</storyboard>
```

Fig. 5. XML Example

VII. Proof of concepts

In order to prove the functioning of the language, we developed an iPad application that allows the creation of storyboards interactively. The system is able to recognize the artist's drawings, convert them to computer language and then performing the disclosed animation.

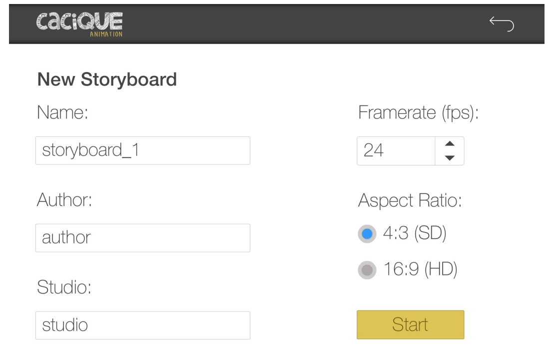
Fig. 6 shows the Cacique Animation app is fully compatible with IOS devices (mainly iPads) and allows users to create their animations by drawing the elements and combining them. The user can preview the animation on any step of the process and edit every element of the language. The video about the app functioning can be seen in the supplementary material.

VIII. Conclusions

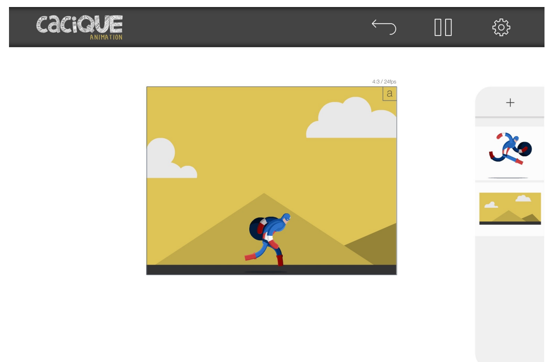
The language presented on this paper is an intent to unify the artist's job and the computer vision software, which allows them to test their animation on real time, and then decide whether to continue or change something.

The main goal of this work is to propose a creativity support framework, composed by a pattern language and a computational tool, which allows artists to perform their work creation process through a digitally-enhanced experience. It is not required from the artist to know any computational language, providing him/her a mechanism that support near real pre-tests during the prototypic creating phases. This solution could also be used to generate simple final animations for advertisement, games, presentations, and so on.

By using industry-adopted standards, different software applications could be built to recognize language symbols and so build automated animations. For instance, mobile applications made with this purpose could make the animation process easier and independent of a more robust hardware - further works points out to this direction.



(a) Storyboard Configuration Screen.



(b) Animatic Screen.

Fig. 6. Cacique Animation

Acknowledgment

The authors would like to thank the University and its PhD Program. This work was partially funded by [ommited for blind revision].

References

[1] P. Wells, *The Fundamentals of Animation*, ser. Fundamentals (Ava) Series. Bloomsbury Academic, 2006. [Online]. Available: http://books.google.com.br/books?id=7sbOu5oEp4oC

[2] S. Cámara, *All about Techniques in Drawing for Animation Production*, ser. All about techniques in. Barron's, 2006. [Online]. Available: http://books.google.com.br/books?id=3yVQAAAAMAAJ

[3] J. Vogt, M. Haesen, K. Luyten, K. Coninx, and A. Meier, "Timisto: a technique to extract usage sequences from storyboards," *EICS '13 Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems*, pp. 113–118, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?id=2494603.2480329

[4] A. Yannopoulos, "DirectorNotation," *Journal on Computing and Cultural Heritage*, vol. 6, no. 1, pp. 1–34, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2442080.2442082

[5] R. Gardies, *Compreender o Cinema e as Imagens*. Armand Colin, 2007.

[6] R. Williams, *The Animator's Survival Kit–Revised Edition: A Manual of Methods, Principles and Formulas for Classical, Computer, Games, Stop Motion and Internet Animators*. Faber & Faber, Inc., 2009.