A Linguistic Formalism for Specifying Visual Representations

Elena Fanea, Sheelagh Carpendale
Department of Computer Science
University of Calgary
{faneael, sheelagh}@cpsc.ucalgary.ca

Abstract

With the proliferation of access to digital media it is becoming increasingly common for people to present information visually. This has led to a myriad of new types of visual representations that frequently come into existence without an associated formalism. It is often difficult to retroactively fit a given formalism to an existing visual representation. We present a formalism that provides us with tools capable of describing visual representations. Using an analogy to natural languages, we build an alphabet composed of two types of ordered letters. With these letters we can develop several languages whose grammar is described by their morphology and syntax. Each language thus defined is capable of describing a family of visual representations. We illustrate this capability by specifying the morphology and syntax necessary to describe two different visual representations of multi-dimensional data, parallel coordinates and glyphs.

Keywords: visual language, linguistic formalism, visual representation, multi-dimensional data, visual alphabet, visual morphology

1. Introduction

It is increasingly common for information to be stored, accessed and exchanged digitally. Since digital media is now more capable of generating and including visuals, the ways in which information is presented are changing. More types of visual representations and visual/textual information integrations are being developed and are in more active use. This includes research fields such as information visualization and informal uses such as the ad hoc inventive use of punctuation in online chat. Horn has declared that combination of all these is in fact a visual language [3]. His definition states that visual language is the integration of words, images and shapes into a single communication unit. While this may seem to be an informal definition of visual language, we have used it as a basis for a linguistic formalism.

In this paper we present a linguistic formalism that can describe visual representations that are created in Information Visualization. Since this formalism has been developed to handle multi-dimensional visual representations we refer to it as Multi-Dimensional Visual Language (MDVL). MDVL consists of an alphabet, $\mathcal{A}$, and specific morphologies and syntax that use this alphabet. Most natural languages that we are familiar with are formalized through an alphabet, which is composed of an ordered list of letters [9]. These letters are grouped according to a selected morphology to form words. Each distinct language has its own morphology. In turn these words are combined according to syntactical rules to create sentences and paragraphs. Our alphabet can be considered an analogy to the Latin alphabet, which is used by English, French and so many other languages. Similarly, our alphabet, $\mathcal{A}$, of MDVL can be used to build several visual languages by defining different morphologies. In this paper we define two such morphologies: a Parallel Coordinates Visual Language (PCVL) and a Glyph Visual Language (GVL). Both PCVL and GVL share the same syntax.

Parallel Coordinates where originally developed by Inselberg [4] and are a powerful and expressive information visualization for multi-dimensional data. However, they are initially difficult to understand and suffer from considerable visual clutter. They have recently received considerable research attention in the form of improved functionally and interactions [12].

Glyphs are also used to visualize multi-dimensional data and have the capability of providing a shape that indicates magnitude differences either for the different attributes of a data item or to reveal the differences in several items across an attribute dimension.

The paper is organized as follows. The next section outlines related research. Section 3 defines our Multi-Dimensional Visual Language (MDVL) alphabet, $\mathcal{A}$, and explains how we have defined letters and created an ordered alphabet. In Section 4 we define a morphology for Parallel Coordinates developing a Parallel Coordinate Visual Language (PCVL). The visual structure of Parallel Coordinates is built up component by component as the morphology is defined. This PCVL is just one example of a visual language that can be built with the MDVL alphabet. A different morphology can be based on another visualization method of multi-dimensional data. To demonstrate this, in Section 5 we will define the morphology for Glyph Visual Language (GVL). Both PCVL and GVL share the same syntax, which is presented in Section 6 and Section 7 concludes the paper.

2. Related Work

Visual languages are defined by Marriott et al. [8] as sets of diagrams that have been defined as valid sentences. They often involve both generative and analytic aspects of formal grammar. Analytic grammars assume that the language has been already generated and analyze whether an arbitrary input string is grammatically correct. They formally describe a parser for a language. It has been suggested that there are three main approaches to the specification of visual languages: grammatical, logical and algebraic [1, 8].

The grammatical approaches are based on string rewrite mechanisms. They have an initial structure, an alphabet and a set of rewrite rules. L-systems [6, 10] are an example of these approaches that can generate complex structures based on rewriting rules. An alphabet, and a set of productions are defined. Productions are the rewriting rules for the individual modules over an interval of time. An L-system development has an initial structure or "axiom" and is mainly used for describing recursive structures.

Rekers and Schürr [11] underline the need to complement the spatial relation graph with an abstract syntax graph. They not only use the graph grammar as syntax definition for formalism for visual languages, but also provide a graphical parsing algorithm for this grammar.

The logical approach uses logic formalisms from mathematics or artificial intelligence. Haarslev [2] is an example that uses artificial intelligence description logic theory to combine topology and spatial relations.

A high-level framework for the definition of visual programming languages is presented in [7]. The layout perspective of the spatial relationships in that formalism is extended to a spatial graph grammar that introduces spatial constraints to the abstract syntax in [5] using algebraic specifications of composing functions to define and compare graphs.

Our formalism relates to the grammatical approaches in that it has an alphabet. In differs in that it has no initial axiom and instead of rewrite rules, we follow a closer analogy to natural language and define morphological units, or words, building a set of available words or vocabulary. In contrast to rewrite rules our words do not necessarily generate from each other. Also, while we use algebraic formalisms we do not rely on the composition of multiple functions to define a grammar. Instead we use spatially located morphological units that relate to each other through spatial location.

3. The Alphabet

Our alphabet is based on two definitions. One, in the Oxford English Dictionary [9] the alphabet is defined as a set of letters or symbols in a fixed order used for writing a language. Two, Horn [3] describes Visual Language as any integration of shapes, images and words but not one of these aspects independently. As a first step we create an ordered alphabet composed of two types of letters, MDletters and PVletters, that integrate shapes and text.

In order to build the alphabet, we will need a set of notations and definitions, to set up the context.

NrDim is the number of dimensions of the dataset, or the number of columns in a data table. Rows usually represent the data item and the columns hold information about the item's dimensions.

NrVisDim is the number of dimensions that are visible at one moment. This needs to be distinct from NrDim because it is possible that not all dimensions will be used in all visualizations.

NrTuples is the number or rows in the table or the number of elements in the dataset.

NrVisTuples is the number of visible elements at one moment.

Notation: *D*=the set of dimensions. *D={j| j=1, NrDim}*

Notation: *visD*=the set of visible dimensions.
visD={j | j = 1, NrVisDim}

Notation: *E*=the set of elements. *E={i | i= 1,NrTuples}*

Notation: *visE*=the set of visible elements.
VisE={i | i = 1, NrVisTuples}

Let *T* be the set of numerical values that we want to visualize, initially stored in the data table and *RGBA* = [0, 1] x [0, 1] x [0, 1] x [0, 1] the set of possible colors, represented by their red, green, blue and alpha components (alpha represents the degree of opacity).

We now have sufficient notation to establish a mapping between the natural language in use (e.g. the data table will be written in a natural language of words and numbers) and the space of the visual language.

Definition 3.1. We define the function *f* as follows:

$$f{:}T \rightarrow (D{\times}E, R^3, RGBA)$$

$$f(P)=(id_{d_P}, id_{t_P}, x_P, y_P, z_P, col_P), \forall P \in T \text{, where:}$$

id_d represents the identification number of the dimension that contains element *P* (in terms of data table, *id_d* is the column number)

id_t represents the identification number of the multi-dimensional item that contains element *P* (in terms of data table, *id_t* is the row number)

x, y, z$\in$ R are the 3D coordinates of the point that represents element *P* in the graphic space. Should one want to create 2D representations *z* can be dormant.

clr is this point's RGBA color, $\text{clr}=(r,g,b,a)$, with $r,g,b,a \in [0,1]$. While it is important that each letter have color, the actual hue, saturation and value of this color is a function of each morphology.

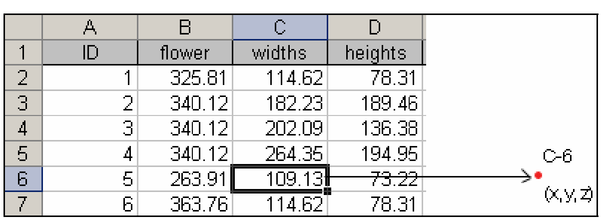
From this it follows that *f* is a well-defined function, which means that each element *P* in the data table has a unique correspondent in the visual representation space.

At this point we have defined a mapping from a multi-dimensional data table to a visual representation space. Each image *f(P)* thus defined is a letter in our aplhabet as follows:

Definition 3.2. We define an **MDletter** as an element $(id_d, id_t, x, y, z, clr) \in (D\times E, R^3, RGBA)$

Notation: $\mathcal{MD}$ = the set of all MDletters = {MDletter} = { $(id_d, id_t, x, y, z, clr) \in (D\times E, R^3, RGBA)$ }.

Therefore an MDletter plots an element *P* from a data table *T* onto a point in a graphic space (Figure 3.1). This point has the coordinates *x, y, z* and is drawn with color *clr*. Detailing how *x, y* and *z* are computed is strictly dependent on the intended visualization technique. Each visualization or family of visualizations will require its own morphology. We define morphology for two visualization techniques in Sections 4 and 5.



	A	B	C	D
1	ID	flower	widths	heights
2	1	325.81	114.62	78.31
3	2	340.12	182.23	189.46
4	3	340.12	202.09	136.38
5	4	340.12	264.35	194.95
6	5	263.91	109.13	73.22
7	6	363.76	114.62	78.31

Figure 3.1: Illustrating the mapping from the multi-dimensional data table to a MDletter in visual representation space

We include *id_t* and *id_d* in the structure of an MDletter for two reasons. First, to comply with Horn's definition of a Visual Language, this combines the graphical point with two labels that state the tuple's identification number, *id_t* and the dimension's identification number, *id_d*.. Second, these two numbers provide a mean of sorting the set of MDletters, completing thus the definition of the alphabet.

We establish the following relation of order on $\mathcal{MD}$.

Definition 3.3. $\forall m_1, m_2 \in \mathcal{MD}$

$$m_1 < m_2 \overset{\text{def}}{\Leftrightarrow} id_{t_1} < id_{t_2} \text{ OR } \left(id_{t_1} = id_{t_2} \text{ AND } id_{d_1} < id_{d_2}\right)$$

$$m_1 = m_2 \overset{\text{def}}{\Leftrightarrow} id_{d_1} = id_{d_2} \text{ AND } id_{t_1} = id_{t_2}$$

Because the MDletters are uniquely identified by their two id numbers, which cannot be bigger than the number of rows and columns in the table, it implies that the set $\mathcal{MD}$ is finite.

Theorem 3.1. ($\mathcal{MD}$, $\le$) is a total relation of order.

Proof: We have to prove that $\forall m_1, m_2 \in \mathcal{MD}$, $(m_1 < m_2)$ OR $(m_1 > m_2)$ OR $(m_1 = m_2)$.

Because $(R, \le)$ is a total relation of order $\Rightarrow (id_{d_1} < id_{d_2})$ OR $(id_{t_1} \ge id_{t_2})$

1) if $id_{t_1} < id_{t_2} \Rightarrow m_1 < m_2$

2) if $id_{t_1} \ge id_{t_2} \Rightarrow$ if $id_{t_1} > id_{t_2} \Rightarrow m_1 > m_2$

if $id_{t_1} = id_{t_2} \Rightarrow$ if $id_{d_1} < id_{d_2} \Rightarrow m_1 < m_2$

if $id_{d_1} > id_{d_2} \Rightarrow m_1 > m_2$

if $id_{d_1} = id_{d_2} \Rightarrow m_1 = m_2$

In order to provide more flexibility and versatility for various visualization methods, we extend the alphabet with a set of special letters that depend only on the set of dimensions, not on the elements of data table. These additional letters can be used to create interaction capabilities such as pivot points. They can be made to be either visible or invisible.

Definition 3.4. We define the function *g* as follows: $g: D \to (D, R^3, RGBA)$

$g(P) = (id_{d_P}, x_P, y_P, z_P, clr_P), \forall P \in D$, where:

id_d represents the identification number of the dimension that contains this element (in terms of data table, id_d is the column number)

x, y, z$\in$ R are the 3D coordinates of the point that represents this element *P* in the graphic space

clr is this point's RGBA color, $\text{clr}=(r,g,b,a)$, with $r,g,b,a \in [0,1]$

Similarly to the above-defined function *f, g* is also a well-defined function. Each image *g(P)* thus defined is a letter in our alphabet as follows:

Definition 3.5. We define a **PVletter** as an element $(id_d, x, y, z, clr) \in (D, R^3, RGB)$.

Notation: $\mathcal{PV}$ = the set of all PVletters = {PVletter} = { $(id_d, x, y, z, clr) \in (D, R^3, RGB)$ }.

PVletters are visualized (or located) by graphical points with coordinates *x, y, z*, color *clr* and one label stating the id_d. A color *clr* is assigned to all PVletters according to morphological rules.

Similar to $\mathcal{MD}$, $\mathcal{PV}$ requires a relation of order that is defined as follows:

Definition 3.6. $\forall p_1, p_2 \in \mathcal{PV},\ p_1 < p_2 \overset{def}{\Leftrightarrow} id_{d_1} < id_{d_2}$, and $p_1 = p_2 \overset{def}{\Leftrightarrow} id_{d_1} = id_{d_2}$.

The above defined relation of order on $\mathcal{PV}$ is also total because it strictly depends on the identification numbers, which are natural numbers, hence totally ordered. Because the set of id numbers of the dimensions is finite, $\mathcal{PV}$ is finite as well.

At this point we have two sets of totally ordered letters. MDletters map a data table element into the visual space and PVletters act as either locations or pivot points in the visual space. The next step is to define the alphabet, which includes both MDletters and PVletters.

Definition 3.7. We define the alphabet, $\mathcal{A}$, as $\mathcal{A} = \mathcal{PV} \cup \mathcal{MD}$.

Definition 3.8. We define a relation of order "<" on $\mathcal{A}$ by extending the relations of order established on $\mathcal{PV}$ and $\mathcal{MD}$ respectively.

$$\forall a_1, a_2 \in \mathcal{A}, a_1 < a_2 \overset{def}{\Leftrightarrow} (a_1, a_2 \in \mathcal{PV} \text{ AND } a_1 < a_2) \text{ OR } (a_1, a_2 \in \mathcal{MD} \text{ AND } a_1 < a_2)$$

$$\forall a_1 \in \mathcal{PV},\ \forall a_2 \in \mathcal{MD} \Rightarrow a_1 < a_2 \tag{3.1}$$

The relation of equality "=" is a natural extension of equality on $\mathcal{PV}$ and $\mathcal{MD}$

Theorem 3.2. $(\mathcal{A}, <)$ is a total relation of order.

Proof: $\forall a_1, a_2 \in \mathcal{A}$ we can have the following situations:

$a_1, a_2 \in \mathcal{PV} \Rightarrow a_1 < a_2$ OR $a_1 \geq a_2$ (Definition 3.6)

$a_1, a_2 \in \mathcal{MD} \Rightarrow a_1 < a_2$ OR $a_1 \geq a_2$ (Theorem 3.1)

$a_1 \in \mathcal{PV},\ a_2 \in \mathcal{MD} \Rightarrow a_1 < a_2$ (Line (3.1)).

Therefore, according to the Oxford Dictionary [9], $\mathcal{A}$ is a well-defined alphabet. To make use of this alphabet we need to define morphologies that are capable of describing how these letters are combined to create visual representations.

4. Morphology for Parallel Coordinates

In this section we will use the alphabet $\mathcal{A}$ to build PCVL and illustrate its descriptive capabilities with Parallel Coordinates. As in any language, the alphabet $\mathcal{A}$ will be used to define morphological units. We called these units words maintaining the analogy to natural language. We start by defining a basic structure for Parallel Coordinates, or a PCword.

Definition 4.1. PCword = $(id_t, (m_1, m_2, \ldots, m_{NrVisDim}), clr)$ an ordered sequence of MDletters, where $id_{t_k} = id_{t_j}, \forall k, j \in VisD$ and $id_{d_k} \neq id_{d_j}, \forall k \neq j, k, j \in VisD$

Since the tuples' id number is common to all components, it becomes the PCword's id_t. The color of m_k is determined by id_{t_k}. Therefore, because all the MDletters of a PCword have the same id_t, all the corresponding points are visualized with the same color (Figure 4-1).

	A	B	C	D
1	ID	flower	widths	heights
2	1	325.81	114.62	78.31
3	2	340.12	182.23	189.46
4	3	340.12	202.09	136.38
5	4	340.12	264.35	194.95
6	5	263.91	109.13	73.22
7	6	363.76	114.62	78.31

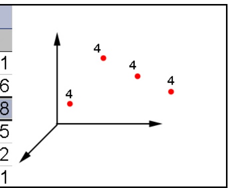


Figure 4-1: In the data table T the tuple 4 is highlighted. This tuple's PCword is drawn in the graphic space on the right. Each MDletter is labeled with its id_t and is colored the same.

Theorem 4.1. In any PCword each visible dimension has a corresponding MDLetter.

Proof: By definition, a PCword $w = (id_t, (m_1, m_2, \ldots, m_{NrVisDim}), clr)$ has MDletters with indices from 1 to NrVisDim. Hence the set of these indices is exactly VisD. We can define a function $f: VisD \to VisD$, $f(k) = id_{d_k}$ where id_{d_k} is the identification number of the dimension corresponding to m_k. From Definition 4.1 $\Rightarrow f(k) \neq f(j), \forall k \neq j, k, j \in VisD$ => f is injective. Because *VisD* is finite => f is injective $\Leftrightarrow$ f is surjective $\Leftrightarrow$ f is bijective. Therefore f is surjective and that means $\forall j \in VisD \Rightarrow \exists! k$ such that $f(k) = j$ $\Rightarrow \forall j \in VisD\ \exists! k \in VisD$ such that m_k corresponds to the j^{th} dimension. => any PCword has one unique letter for each visible dimension.

Definition 4.2. $p: VisD \to VisD$, p bijective. We say that *p* is a **permutation** of *VisD*.

Notation: Let p be a permutation of VisD. Then a PCword $w = (id_t, (m_1, m_2, \ldots, m_{NrVisDim}), clr)$ where id_{d_k} is the dimension corresponding to m_k can be written $\mathbf{(id_t, m_{p(1)} m_{p(2)} \ldots m_{p(NrVisDim)}, clr)}$ given that $p(k) = id_{d_k}$.

Notation: $\mathcal{PCW}$ = the set of all PCwords = { $(id_t, m_{p(1)} m_{p(2)} \ldots m_{p(NrVisDim)}, clr)$ | $m_{p(k)} \in \mathcal{MD}$, k=1, NrVisDim, id_t= 1, NrVisTuples, p= permutation of VisD}.

As we can observe, a PCword corresponds to one tuple in Parallel Coordinates, which visualizes one row of the data table. A PCword is a morphological unit based solely on $\mathcal{MD}$, but using the rest of the alphabet is essential for a proper formal description. Next we define another type of word, which uses elements of $\mathcal{PV}$.

Definition 4.3. PVword = $((v_1, v_2, \ldots, v_{NrVisDim}), clr)$ an ordered sequence of PVletters, where $id_{d_k} \neq id_{d_j}, \forall k \neq j, k,j \in VisD$.

The color used to visualize a PVword is the same used for each component v_k (Figure 4-2).

Notation: Let p be a permutation of VisD. Then an PVword vw = $((v_1, v_2, \ldots, v_{NrVisDim}), col)$ where id_{d_k} is the dimension corresponding to m_k can be written $\mathbf{(v_{p(1)}v_{p(2)}\ldots v_{p(NrVisDim)}, clr)}$, where $p(k) = id_{d_k}$.

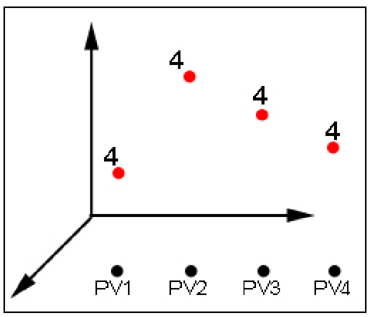


Figure 4-2 A PCword and the corresponding PVword

Theorem 4.2. For each permutation p of VisD there is only one PVword.

Proof: Similar with the proof of Theorem 4.1 we can show that all the visible dimensions are represented by a PVletter in a PVword. But we have only NrVisDim PVletters available, hence the sole difference between any two PVwords consists in the order the PVletters are used.

Notation: $\mathcal{PVW}$ = $\{(v_{p(1)}v_{p(2)}\ldots v_{p(NrVisDim)}, clr) \mid v_{p(k)} \in \mathcal{PV}$, k=1, NrVisDim, p= permutation of VisD$\}$ the set of all PVwords.

PCwords and PVwords form the main morphological units. However, in themselves they are not sufficient to build the Parallel Coordinate visual representation. Next we define a PCline that together with a PCword, formally describes a tuple in Parallel Coordinates.

Definition 4.4. Let w = $(id_{tw}, m_{p(1)}m_{p(2)}\ldots m_{p(NrVisDim)}, clr_w)$ be a PCword, where p is a permutation of VisD. We define a **PCline of the PCword w** = $(id_{tw}, \bigcup_{i=1}^{NrVisDim-1} [m_{p(i)}, m_{p(i+1)}], clr_w)$.

The PCword's id_t and color will be assigned to the corresponding PCline. Graphically a PCline (Figure 4-3) is the poly-line that connects all the points that visualize the MDletters composing the PCword (Figure 3).

Notation: $\mathcal{PCL}$ =the set of all PClines = { PCline of the PCword w | w $\in \mathcal{PCW}$ }.

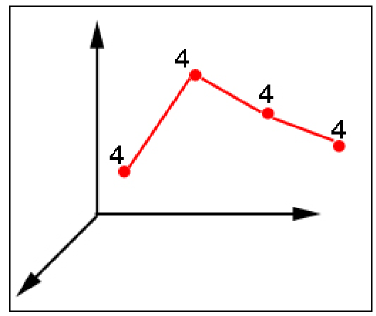


Figure 4-3: A PCline links the MDletters of a PCword

In order to fully formalize the Parallel Coordinates, we need means to describe the other elements of this visual representation. The first is the line that connects the pivot points.

Definition 4.5. Let vw = $(v_{p(1)}v_{p(2)}\ldots v_{p(NrVisDim)}, clr)$ be a PVword, where p is a permutation of VisD. We define a **PVline for PVword vw** = $(\bigcup_{i=1}^{NrVisDim-1} [v_{p(i)}, v_{p(i+1)}], clr)$.

Similar to a PCline, a PVline is the poly-line that connects the points representing the PVletters that compose the PVword and it is visualized with the same color as the Pvletters (Figure 4-4). From Theorem 4.2 it follows that, for a given permutation p of VisD, there is only one PVline

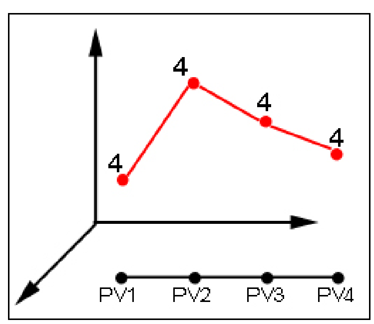


Figure 4-4 A PCline and the corresponding PVline

In Parallel Coordinates the pivot line is connected with the tuples and the points representing elements of the data table through a set of parallel axes. Translated into PCVL, this set of axes is defined as follows:

Definition 4.6. Let $w = (id_{tw}, m_{p(1)}m_{p(2)}\ldots m_{p(NrVisDim)}, clr_w)$ be a PCword and $vw = (v_{p(1)}v_{p(2)}\ldots v_{p(NrVisDim)}, clr)$ be a PVword, where p is a permutation of VisD. We define a **PCax of the PCword w =** $(\bigcup_{i=1}^{NrVisDim} [v_{p(i)}, m_{p(i)}], clr_{ax})$.

Graphically, a PCax (Figure 4-5) is the set of segments that connect each point representing an MDletter of the PCword with the corresponding PVletter of the PVword. The color clr_{ax} is common to all PCaxes, but independent of other units' color.

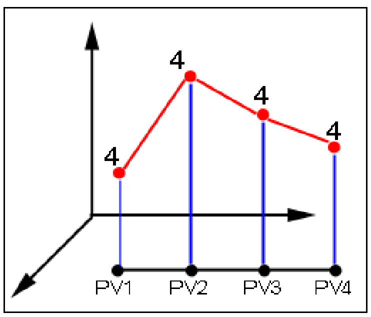


Figure 4-5 Parallel coordinates representation showing a PCax that connects the MDletters of a PCline with the corresponding PVletters of the PVline.

Notation: $\mathcal{PCAx}$ = the set of all PCaxes ={ PCax of the PCword w | w $\in \mathcal{PCW}$ }.

5. Morphology for Glyphs

Glyphs based on a data table are independent visual representations and are usually created either one per row to create a characteristic shape indicative of the tuple's properties or one per column to show comparative magnitude for one dimension across the tuples. The Glyphs formalized here are of the letter variety. Defining a morphology for Glyphs begins with the basic structure for Glyphs, or a Gword. A Gword is an analogue morphological unit to a PCword that it visualizes one column in contrast to one row.

Definition 5.1. Gword = $(id_d, (m_1, m_2, \ldots, m_{NrVisTuples}), clr)$ an ordered sequence of MDletters, where $id_{d_k} = id_{d_j}, \forall k,j \in$ VisE and

$id_{t_k} \neq id_{t_j}, \forall k \neq j, k,j \in$ VisE.

Because we are defining words for both Parallel Coordinates (PCVL) and Glyphs (GVL), we establish coherent color policy. Therefore since the MDletters that form a Gword have different id_t they will have different colors. The Gword's color *clr* is then an array with all the components' colors (Figure 5-1).

	A	B	C	D
1	ID	flower	widths	heights
2	1	325.81	114.62	78.31
3	2	340.12	182.23	189.46
4	3	340.12	202.09	136.38
5	4	340.12	264.35	194.95
6	5	263.91	109.13	73.22
7	6	363.76	114.62	78.31

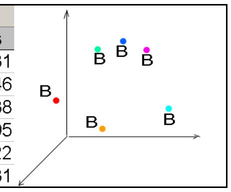


Figure 5-1: On the left, the Gword's data dimension column; on the right, the Gword's MDletters, labeled according to their column and colored uniquely according to their id_t.

Theorem 5.1. In any Gword each visible tuple has a corresponding MDLetter.

Proof: Analog to the proof of Theorem 4.1.

Notation: Let q be a permutation of VisE. Then an Gword $gw = (id_d, (m_1, m_2, \ldots, m_{NrVisTuples}), clr)$ where id_{t_k} is the tuple corresponding to m_k can be written $\mathbf{(id_d, m_{p(1)}m_{p(2)}\ldots m_{p(NrVisTuples)}, clr)}$, where $q(k)=id_{t_k}$.

Notation: $\mathcal{GW}$ = the set of all Gwords = $\{m_{p(1)}m_{p(2)}\ldots m_{p(NrVisTuples)} \mid m_{p(k)} \in \mathcal{MD}$, k=1, NrVisTuples, p= permutation of VisE}.

At this point we include in GVL one of PCVL's morphological units, PVwords. The auxiliary morphological units necessary to completely formalize the Glyphs are defined as follows.

Definition 5.2. Let $gw = (id_{d_{gw}}, m_{p(1)}m_{p(2)}\ldots m_{p(NrVisTuples)}, clr)$ be a Gword, where p is a permutation of VisD. Let v be the PVletter that corresponds to dimension $id_{d_{gw}}$. We define a **Gfan of the Gword gw =** $(id_{d_{gw}}, \bigcup_{i=1}^{NrVisTuples} [v, m_{p(i)}], clr_f)$ where the Gword's $id_{d_{gw}}$ is also used by the Gfan.

Graphically, a Gfan is the set of segments that connect the pivot point corresponding to $id_{d_{gw}}$ with each of the points that visualize the MDletters of the Gword. These segments are visualized with the same color col_f, independent of the individual colors of the MDletters that determine the Gfan (Figure 5-2).

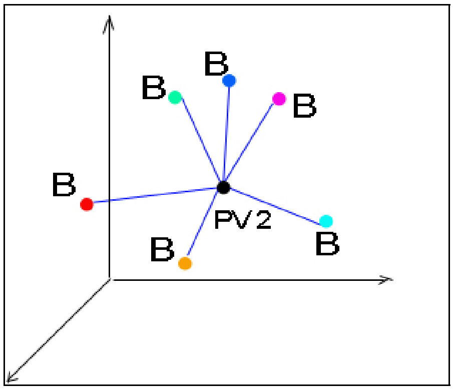


Figure 5-2: An emerging Glyph showing its centre, PV2, its multi colored Gword and its Gfan.

Notation: $\mathcal{GF}$ = { Gfan of the Gword gw | gw $\in \mathcal{GW}$ } the set of all Gfans.

Definition 5.3. Let gf =($id_{d_{gf}}$, $\bigcup_{i=1}^{NrVisTuples} [v, m_{p(i)}]$, clr_f) be a Gfan of the Gword ($id_{d_{gf}}$, $m_{p(1)} m_{p(2)} \dots m_{p(NrVisTuples)}$, clr). We define a **Gglyph of the Gword gw** = ($id_{d_{gf}}$, $\bigcup_{i=1}^{NrVisTuples-1} \triangle v m_{p(i)} m_{p(i+1)}$, clr_g).

Naturally, the Gglyph has the same $id_{d_{gf}}$ as the Gfan and the MDletters of the Gword.

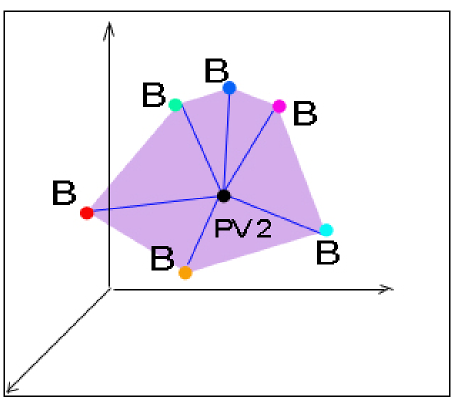


Figure 5-3: A Gglyph

Graphically, the Gglyph represents the surface obtained as the union of all filled-triangles, each triangle having the corresponding pivot point, In Figure 5-3 the PVword PV2, as one vertex and pairs of points that visualize consecutive MDletters as the other two vertices

Notation: $\mathcal{GG}$ = the set of all Gglyphs ={ Gglyph of the Gword gw | gw $\in \mathcal{GW}$ }.

Definition 5.4. Let gg be a Gglyph of the Gword gw = ($id_{d_{gf}}$ $m_{p(1)} m_{p(2)} \dots m_{p(NrVisTuples)}$, clr_g), gg=($id_{d_{gf}}$, $\bigcup_{i=1}^{NrVisTuples} \triangle v m_{p(i)} m_{p(i+1)}$, clr_g). We define the **Gborder of the Gglyph gg** = ($id_{d_{gf}}$, $\bigcup_{i=1}^{NrVisTuples-1} [m_{p(i)}, m_{p(i+1)}]$, clr_b).

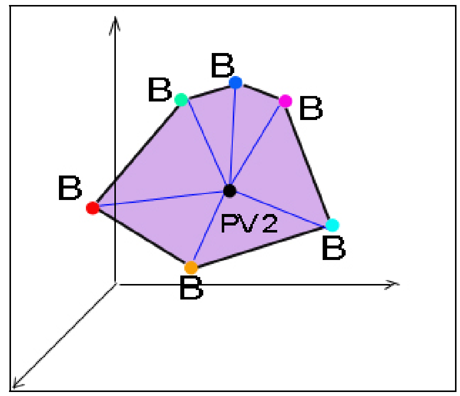


Figure 5-4: A complete Glyph with its Gglyph and Gborder

Similar to the Gglyph, the Gborder keeps the same $id_{d_{gf}}$. The Gborder represents the poly-line that connects all the points that visualize consecutive MDletters of the Gglyph (Figure 5-4).

Notation: $\mathcal{GB}$ = the set of all Gborders ={ Gborder of the Gglyph gg | gg $\in \mathcal{GG}$ }.

6. Syntax

An important aspect for syntax of a visual representation is the maintenance of topology. We define acceptable transformation as those that maintain topology or properties of the geometric configurations as unaltered by elastic deformations such as a stretching or a twisting. In the topology of MDVL we consider the coordinates of the morphological units because they form the basis for the visible shapes. Geometric transformations of coordinates are allowed only if they preserve the structures and the meanings of these units.

The MDVL grammar is at this point analytic. From a multi-dimensional data table, the data is visualized according to set of corresponding MDVL sentences.

Definition 6.1. The **MDVL topology** is the geometric configuration of the morphological units, preserved by a specific set of permissible transformations.

Definition 6.2. The **MDVL syntax** is the set of rules that describe the conditions under which a sentence is grammatically-correct in a given topology.

Axiom 6.1. Any morphological unit is used once and only once in a sentence.

Axiom 6.2. The only geometric transformations allowed on the morphological units are:

1) Scaling

The only scaling allowed is on the x direction. It can be defined on both $\mathcal{PV}$ and $\mathcal{MD}$ sets. Let $w=(id_{d_w}, x_w, y_w, z_w, col) \in \mathcal{PV}$ and $S_x: \mathcal{PV} \times R \rightarrow \mathcal{PV}$ $S_x(w,\alpha)=w'$, where $w'=(id_{d_{w'}}, x_{w'}, y_w, z_w, col) \in \mathcal{PV}$, $x_{w'}=\alpha * x_w$. Similarly we define scaling for $\mathcal{MD}$. Let $w=(id_{t_w}, id_{d_w}, x_w, y_w, z_w, col) \in \mathcal{MD}$, $S'_x: \mathcal{MD} \times R \rightarrow \mathcal{MD}$ $S'_x(w,\alpha)=w'$, $w'=(id_{t_w}, id_{d_{w'}}, x_{w'}, y_w, z_w, col) \in \mathcal{MD}$, $x_{w'}=\alpha * x_w$. These transformations can be easily extended to $\mathcal{PVW}$, and $\mathcal{MDW}$ by applying the scaling to each component.

2) Rotations
3) Translation

Analog to scaling, rotations and translations in all three directions can be defined for all morphological units.

7. Conclusions

In this paper we have presented a formal approach to description of visual representations using an analogy to natural languages. We have defined an alphabet, MDVL, consisting on two types of ordered letters that can be used as the basis for the development of several languages. Two examples illustrate the way the description of a family of visual representations can be based on this alphabet with: we have elaborated the morphology and the syntax for two visual representations of multi-dimensional data, parallel coordinates and glyphs.

Our linguistic formalism of visual representations extends the influence of Chomsky grammars from visual programming languages to information visualization techniques. The approach we have proposed here provides a theoretical foundation for description of visual representations, which can be further investigated for other techniques than those detailed here.

Acknowledgments

We gratefully acknowledge support from Natural Sciences and Engineering Research Council (NSERC) and thank our colleagues at Interactions Lab, University of Calgary, for their insightful comments.

8. References

[1] Costagliola, Gennaro, Delucia, Andrea, Orefice, S. and Polese, Giuseppe, "A Classification Framework to Support the Design of Visual Languages", *Journal of Visual Languages and Computing*, vol. 13, nr. 6, 2002, pp. 573-600

[2] Haarslev, Volker, "A Fully Formalized Theory for describing Visula Notations", in Marriott, Kim and Meyer, Bernd editors, *Theory of Visual Languages,* Springer-Verlag New-York, 1998, pp. 261-292

[3] Horn, Robert, *Visual Language-Global Communication for the 21st Century*, MacroVu, Inc, Bainbridge Island, Washington, 1998

[4] Inselberg, Alfred and Dimsdale, Bernard., "Parallel Coordinates: A Tool for Visualizing Multi-dimensional Geometry", *IEEE Conference on Visualization 1990,* IEEE CS Press , pp. 361-378

[5] Kong, Jun and Zhang, Kang, "On a Spatial Graph Grammar Formalism", *IEEE Symposium on Visual Languages and Human-Centric Computing 2004*, IEEE CS Press, pp. 102-104

[6] Lindenmayer, Aristid, "Mathematical models for cellular interaction in development, Parts I and II", *Journal of Theoretical Biology,* vol, *18* 1968, pp. 280–315

[7] Marriott, Kim, "Constraint Multiset Grammars", *IEEE Symposium on Visual Languages 1994*, IEEE CS Press, pp. 118-125

[8] Marriott, Kim, Meyer, Bernd, and Wittenburg, Kent B., "A Survey of Visual Language Specification and Recognition", in Marriott, Kim and Meyer, Bernd editors, *Theory of Visual Languages,* Springer-Verlag New-York, 1998, pp. 5-85

[9] *Oxford Advanced Learner's Dictionary of Current English*, Oxford University Press, 2000

[10] Prusinkiewicz, Przemyslaw, "Simulation Plants and Plant Ecosystems", *Communications of the ACM,* vol. 43, nr. 7, 2000, pp. 84-93

[11] Rekers, J., and Schrr, A., "Defining and Parsing Visual Languages with Layered Graph Grammars", *Journal of Visual Languages and Computing*, vol. 8, nr. 1, 1997, pp. 27-55

[12] Yang, Jing; Peng, Wei; Ward, Mathew. O. and Rundensteiner, Elke A., "Interactive Hierarchical Dimension Ordering, Spacing and Filtering for Exploration of High Dimensional Datasets", *IEEE Symposium on Information Visualization 2003*, IEEE CS Press, pp. 105-112