Revised July, 1995
This paper appeared in
J. of Comb. Maths. and Comb. Computing 19 (1995), pp 171-191.

Some Algorithms for the Computer Display of Geometric Constructions in the Real Projective Plane

William Kocay* and Don Tiessen
Computer Science Department
University of Manitoba
Winnipeg, Manitoba, CANADA, R3T 2N2
e-mail: bkocay@cs.umanitoba.ca

Abstract

Several algorithms for geometric constructions on the real projective plane are described. These methods also apply to euclidean plane geometry. The concept of an augmented determining set is fundamental to the algorithms. A backtracking algorithm to find augmented determining sets is described. Algorithms for animating constructions, and an incidence-forcing algorithm are also presented. These algorithms have been implemented on an X-Windows system.

1. Determining Sets

We will be working with synthetic constructions in the real projective plane. Since the real affine plane is a subset of the projective plane, the results will also hold for constructions in the euclidean plane. We refer the reader to the books by Coxeter [2] and Pedoe [4] as references for synthetic projective geometry. Consider the geometric construction of Figure 1. It contains a line ℓ with four points A, B, C, and D. If A, B, and C are chosen arbitrarily on ℓ, then the diagram gives the construction for finding the harmonic conjugate D of point C with respect to A and B.

* This work was supported by an operating grant from the Natural Sciences and Engineering Research Council of Canada.

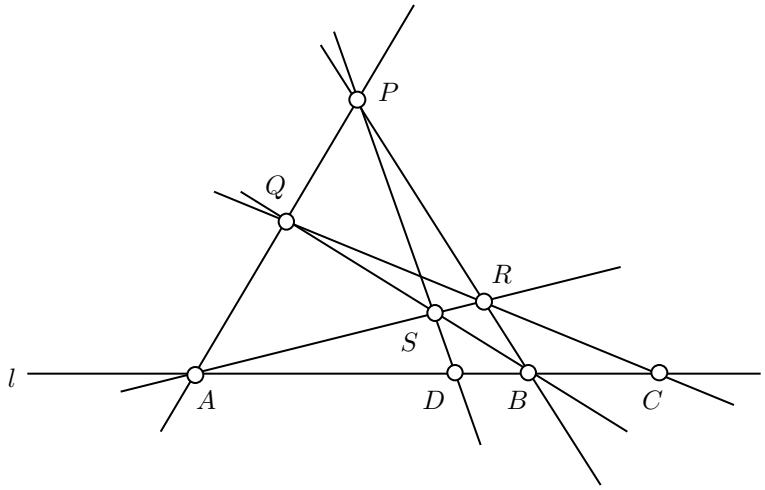

Figure 1 Harmonic conjugates

Suppose that this drawing has been constructed on a computer screen. The user now selects a point P, say, with the intention of moving it slightly. This means that the lines through P must move with P and consequently the points Q, R, and S must also move slightly. All point-line incidences in the construction are to be maintained while P moves. However, no matter where P is moved to, we can require that A and B remain fixed. C and D will then move in harmonic ratio with respect to each other, since they are harmonic conjugates relative to A and B, and this is independent of the particular point P used to construct them.

In this paper algorithms are presented which allow any point or line of such constructions to be moved, thereby animating the entire configuration. Current geometry software does not seem to support this kind of general movement. The algorithms have been implemented on an X-Windows system, and are found to work well.

In general, any construction in the projective plane can be represented by an ordered pair (Σ, Π), where Σ is a set of points and lines and Π is the set of incidences between them. We will refer to a pair (Σ, Π) as a *geometric configuration*. For example, a triangle with points A, B, C and lines a, b, c can be represented by the pair $(\{A, B, C, a, b, c\}, \{Ab, Ac, Ba, Bc, Ca, Cb\})$. This representation is abstract in the sense that no drawing of the configuration need be given. If we do want to draw the construction, we need to assign positions to the lines and points. The positions must be assigned so that the incidences in Π are maintained. If we know the position of points A, B, and P in Figure 1, this determines the lines of

the triangle ABP. If we also know the position of S, this determines the lines through S, which in turn determines the points Q, R, and D, which in turn determines the line QR, which then determines C. This illustrates the concept of *determined* objects.

Notice that a configuration (Σ, Π) can be viewed as a bipartite incidence graph of points versus lines. However not every bipartite graph corresponds to a geometric configuration.

1.1 Definition. Let (Σ, Π) be a geometric configuration, $S \subseteq \Sigma$ be a set of points and lines. Let $S_0 = S$. For each $i \geq 1$, we define

$$\begin{aligned} S_i = S_{i-1} &\cup \{P \mid P \text{ is incident with exactly 2 lines of } S_{i-1}\} \\ &\cup \{\ell \mid \ell \text{ is incident with exactly 2 points of } S_{i-1}\} \end{aligned}$$

Let $S^* = S_j$, where $j = \min\{i \mid S_i = S_{i+1}\}$. All objects $o \in S^*$ are said to be *determined* by S. The *rank* of an object $o \in S^*$ with respect to S is $r(o) = \min\{i \mid o \in S_i\}$. If an object has rank i, then it is said to be determined by S_{i-1}.

1.2 Definition. Given a configuration (Σ, Π), a *determining set* for this construction, denoted by $\Delta(\Sigma, \Pi)$, is defined as a minimal subset of Σ that determines *all* of the objects in Σ, such that *no two objects of the same rank are incident*; that is, $\Delta(\Sigma, \Pi)$ determines all of Σ, but no proper subset of Δ does.

Thus if the positions of the points and lines in Δ are known, then the positions of all objects in Σ are known. Some configurations do not have determining sets. We shall come back to this point later. Note the restriction that no two objects of the same rank be incident. This is to prevent the situation where two independently determined objects are required to be incident, which in general may not be possible.

In the construction of Figure 1, we can take $\Delta = \{A, B, P, S\}$. These objects have rank 0. These four points determine six lines, which all have rank 1. The intersections of these six lines determine the points Q, R, and D, which have rank 2. The line QR is then determined with rank 3. Finally the point C is determined, with rank 4. Notice that the lines QD, RD, PC, etc., could also be drawn, but they are *not part* of the configuration (Σ, Π).

This example illustrates an important point of the definition. An object o of rank i is determined by *exactly two* objects of rank $i-1$. These objects are called its *antecedents* and are denoted by $a_1(o)$ and $a_2(o)$. Since the incidence graph is bipartite, a point is determined by two lines, and a line is determined by two points.

1.3 Lemma. Given a determining set $\Delta(\Sigma, \Pi)$, we can arbitrarily assign the positions of the objects in Δ. The positions of all remaining objects of Σ are then uniquely determined.

Proof. Assign the positions of the objects of Δ arbitrarily. These are the objects of rank 0. There are no incidences among them. The objects of rank 1 are each determined by exactly two objects of rank 0. Therefore their positions are uniquely determined. Then the objects of rank 2 are assigned positions, and so on, until all objects of Σ have been assigned positions.

Thus if we move point P in the configuration of Figure 1, and require that A, B, and S remain fixed, then the positions of these four points are completely known. We can then compute the positions of all remaining objects in the configuration, and update the diagram in real time, as point P is moved.

Homogeneous Coordinates

We assume that the positions of points and lines are stored as homogeneous coordinates in the real projective plane. Positions of points will be represented by triples (x, y, z). Positions of lines will also be represented by triples $[x, y, z]$. The coordinates will be stored as floating point numbers. Point $P = (P_x, P_y, P_z)$ and line $m = [m_x, m_y, m_z]$ are incident if and only if $P \cdot m = 0$. If $P = (P_x, P_y, P_z)$ and $Q = (Q_x, Q_y, Q_z)$ are two points, then the line $\ell = PQ$ can be computed as the cross product $P \times Q$ of the coordinate vectors. Similarly the intersection of two lines $m = [m_x, m_y, m_z]$ and $n = [n_x, n_y, n_z]$ can be computed as the cross product $m \times n$ of the coordinate vectors. We shall denote an object and its homogeneous coordinate vector by the same symbol. Thus if ℓ and m are two lines, and P is their point of intersection, we shall write $P = \ell \times m$. The cross product operation tends to produce very large coordinates after a while. Therefore from time to time it is necessary to renormalize them by dividing by some non-zero value α. This is permissible, since the homogeneous coordinates $(x/\alpha, y/\alpha, z/\alpha)$ and (x, y, z) represent the same object. It is often convenient to take $\alpha = \max\{|x|, |y|, |z|\}$.

In order to display a configuration on the computer screen, we have to map the homogeneous coordinates to euclidean plane coordinates. There are several ways of doing this. We can select any coordinate, say the z-coordinate. Given a point $P = (x, y, z)$, if z is non-zero we divide by z to produce the cartesian coordinates $(x/z, y/z)$. This is equivalent to normalizing (x, y, z) to make the third coordinate 1. If $z = 0$, then (x, y, z) is a point at infinity, and we cannot draw it in the xy-plane. Since we are using floating point numbers, we have to replace the condition $z = 0$ with an inequality $|z| < \epsilon$, where ϵ is a suitable small constant. A line $\ell = [a, b, c]$ is mapped to the equation $ax + by + c = 0$ in the cartesian plane, since $P \cdot \ell = 0$ and we have normalized z to 1. We can obtain different views of the same configuration by using a different coordinate for normalization. Normalizing on the first or second coordinates will produce

different screen drawings of the same configuration. Different points will appear to be at infinity. We can also implement zooming by normalizing a coordinate to some other value than 1. Panning across the projective plane can also be implemented by a suitable linear transformation of the homogeneous coordinates. We can even pan all the way to points at infinity. Thus points at infinity are no different from other points. Any point can be considered as a point at infinity.

Suppose that an object o' in a configuration (Σ, Π) is to be moved. Once a determining set Δ has been constructed such that $o' \in \Delta$, we can animate the diagram. We store the objects of Σ on a queue Q according to their rank. Q is an array of length $n = |\Sigma|$. For each object $o = Q[i]$ $(i = 1, \ldots, n)$, we execute the following steps.

- If $o \in \Delta$, then either $o = o'$, in which case the position of o is determined by the movement applied; or else $o \neq o'$, in which case the position of o is unchanged.
- If $o \notin \Delta$, then o has two antecedents $a_1(o)$ and $a_2(o)$ also on the queue. The new coordinates of o are defined as $o = a_1(o) \times a_2(o)$.

General Position

Consider a determining set $\Delta(\Sigma, \Pi)$. If $A, B, C \in \Delta$, then A, B, and C cannot be collinear on a line ℓ, since ℓ would be incident with three objects of Δ, which contradicts the definition of determined. Similarly no three lines $\ell, m, n \in \Delta$ can be concurrent. It is common to say that a set of points in the plane is in *general position* if no three of them are collinear. Similarly, a set of lines in the plane is in *general position* if no three of them are concurrent. The concept of a determining set is a generalization of the notion of general position. The difference is that general position refers to the positions of only lines or only points. There is one further difference, as indicated by the following definition.

1.4 Definition. Consider a configuration (Σ, Π). A set of points $P_1, P_2, \ldots, P_n \in \Sigma$ is *collinear with respect to* (Σ, Π) if there is a line $\ell \in \Sigma$ with incidences $P_1\ell, P_2\ell, \ldots, P_n\ell$. Notice that points which are collinear in the plane need not be collinear with respect to (Σ, Π). Similarly a set of lines $\ell_1, \ell_2, \ldots, \ell_n \in \Sigma$ is *concurrent with respect to* (Σ, Π) if there is a point $P \in \Sigma$ with incidences $P\ell_1, P\ell_2, \ldots, P\ell_n$. Lines which are concurrent in the plane need not be concurrent with respect to (Σ, Π).

Determining Sets and Incidence Graphs

Given a configuration (Σ, Π), let $G(\Sigma, \Pi)$ denote its incidence graph. Σ is the vertex set of G and Π is the edge set. Let Δ be a determining set. The nodes of G can be partitioned according to their rank. There are no edges

connecting two objects of the same rank; therefore every edge connects two objects of different rank. The incidence graph of Figure 1 is shown in Figure 2 with the rank of each object indicated.

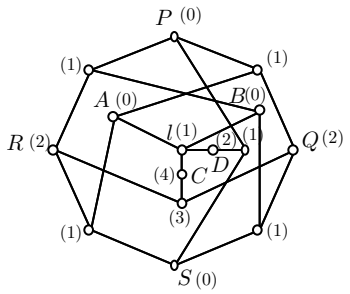

Figure 2 Incidence graph of harmonic conjugates

1.5 Proposition. *Given a determining set $\Delta(\Sigma, \Pi)$, the incidence graph $G(\Sigma, \Pi)$ has an even number of edges. Let E be the number of edges of G, let $n = |\Sigma|$ and $k = |\Delta|$. Then $k = n - E/2$.*
Proof. Each object of rank $i \geq 1$ is adjacent to exactly two objects of rank less than i. This accounts for all edges of G. Since the k objects of Δ have rank zero, it follows that $E = 2(n - k)$, as required.

1.6 Corollary. *All determining sets of (Σ, Π) have the same cardinality*.

The number of objects in a determining set $\Delta(\Sigma, \Pi)$ is called the *dimension* of (Σ, Π). The harmonic conjugates configuration of Figure 1 has dimension 4. In general, if G does not have an even number of edges then a determining set does not exist. As we shall see later, an even number of edges is not a sufficient condition for a determining set to exist.

An Algorithm to Find Determining Sets

The algorithm that we have used to find determining sets is an exhaustive backtrack search. Assume that there are n objects in the configuration, and that the objects are numbered 1,2, $\ldots, n$. We use two queues, $Q1$ and $Q2$, stored as arrays $(1 \ldots n)$. $Q2$ contains the determined objects, that is, it contains the elements of the determining set, as well as all objects incident with exactly *two* previously determined objects. When an undetermined object o is found to be incident with a determined object, o is placed on $Q1$. When an object $o \in Q1$ is found to be incident with another determined object, o thereby becomes determined; so it is placed on $Q2$, as well. Thus $Q1$ contains all objects incident with one or two previously determined objects. We have two procedures, *DetermineObjects* and *FindDetSet*. *DetermineObjects*($Q2$) is called after a new object o has been

placed on $Q2$. It performs a breadth-first search of the incidence graph, beginning at o, finding all objects incident with o, and placing them on $Q1$ and/or $Q2$, until no new determinined objects can be found. If at any point it finds an object incident with three previously determined objects, it returns *false*. Otherwise it returns *true* to indicate success. *DetermineObjects* is a straightforward breadth-first search, and we do not show its pseudo-code. *FindDetSet*(o) is shown below. It places object o on $Q2$ and then calls itself recursively for each object $o' > o$ until either a determining set is found, or until all possibilities have been exhausted. We assume that the objects have been numbered $1 \ldots n$ in some (arbitrary) order.

```
FindDetSet(o: object): boolean
{ extend the current determining set to include object o }
{ returns true if a determining set is found }
begin
  save the current sizes of Q1 and Q2
  add o to Q2
  if DetermineObjects(Q2) then begin
    if Q2 contains all n objects then return(true)
    for o' := o + 1 to n do if o' ∉ Q1 then
      { try to add o' to the determining set }
      if FindDetSet(o') then return(true)
  end
  { either DetermineObjects returned false or else all o' were tried }
  restore Q1 and Q2
  return(false)
end   { FindDetSet }
```

When the algorithm terminates, the objects of $Q2 - Q1$ form a determining set Δ, if one exists. For each $o \in Q2 - \Delta$, the two antecedents $a_1(o)$ and $a_2(o)$ are also found. They are stored by the procedure *DetermineObjects*. We save these antecedents and the queue $Q2$. When one of the objects in Δ is moved, the diagram can be animated by scanning $Q2$ from head to tail and computing the new coordinates of each object from its antecedents. This takes a constant number of steps per object. If there are n objects in total, then $O(n)$ steps are required to update the diagram when a point or line is moved. When an object o is to be moved we require a determining set Δ containing o. In order to find Δ, we initialize the queue $Q2$ to contain o before calling *FindDetSet*. This requires a small change in the code.

2. Augmented Determining Sets

Figure 3 shows a construction consisting of two triangles ABC and $A'B'C'$ in perspective from a point P. (The dotted line ℓ is not part of the configura-

tion.) By Proposition 1.5, this configuration has no determining set, since its incidence graph has 21 edges. We introduce augmented determining sets in order to deal with such configurations. Augmented determining sets are similar to determining sets, except that they can contain constrained objects.

2.1 Definition. Let (Σ, Π) and (Σ', Π') be configurations where $\Sigma' \subseteq \Sigma$ and $\Pi' \subseteq \Pi$. (Σ', Π') is an *induced* configuration if, whenever there is an incidence $\ell P \in \Pi$, where $\ell, P \in \Sigma'$, then $\ell P \in \Pi'$. This means that the incidence graph of (Σ', Π') is an induced subgraph of the incidence graph of (Σ, Π).

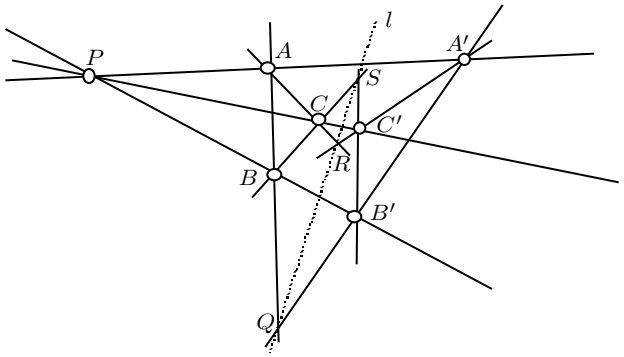

Figure 3 Two triangles in perspective

Now let (Σ, Π) be a configuration that has no determining set. *FindDetSet* will construct sets $S \subseteq \Sigma$ such that S is a determining set for some induced configuration (Σ', Π'). Although S does not determine the entire construction, it determines $S^* = \Sigma'$. Let o be an object of $\Sigma - S^*$ which is incident with *exactly one* object $m \in S^*$, if there is such an o. The object o is said to be *constrained* by m. We now add o to S^* and compute $(S^* \cup \{o\})^*$, which will determine more of Σ. We modify *FindDetSet* by placing the following for-loop immediately before the final return(false) statement. This loop saves the current size of $Q1$, which contains all constrained objects, in a *global* variable *Q1Size*, and then tries to add each constrained object $o \in Q1$ to $Q2$.

```
Q1Size := |Q1|   { save current size of Q1 }
for k := 1 to Q1Size do begin
  o := Q1[k]   { k^th object on Q1 }
  if o ∉ Q2 then if AugDetSet(o) then return(true)
end
```

This calls a procedure *AugDetSet*(o), which adds the constrained object o

to $Q2$ and tries to extend $Q2$ with *constrained objects* up to $Q1[Q1Size]$ until every object is determined.

```
AugDetSet(o: object): boolean
{ add the constrained object o to the determining set }
{ returns true if an augmented determining set is found }
begin
  save the current sizes of Q1 and Q2
  k := position of o on Q1
  add o to Q2
  if DetermineObjects(Q2) then begin
    if Q2 contains all n objects then return(true)
    for k := k + 1 to Q1Size do begin
      o' := Q1[k]   { k^th object on Q1 }
      if o' ∉ Q2 then if AugDetSet(o') then return(true)
    end
  end
  { either DetermineObjects returned false or else all o' were tried }
  restore Q1 and Q2
  return(false)
end   { AugDetSet }
```

If the algorithm is successful, it will terminate with all n objects on $Q2$. $Q2$ will contain a set Δ_0 of undetermined objects and a set Δ_1 of constrained objects for which $(\Delta_0^* \cup \Delta_1)^* = \Sigma$. Each constrained object $o \in \Delta_1$ is incident with exactly one object of Δ_0^*. We call $\Delta^+ = (\Delta_0, \Delta_1)$ an *augmented determining set*. This algorithm suggests the following definition of augmented determining set.

2.2 Definition. An *augmented determining set* Δ^+ for a configuration (Σ, Π) consists of a set Δ_0 and a set Δ_1 such that:

i) Δ_0 is a determining set for an induced configuration (Σ', Π');
ii) $(\Delta_0^* \cup \Delta_1)^* = \Sigma$;
iii) each $o \in \Delta_1$ is incident with exactly one object in Σ';
iv) Δ_1 is a minimal set with properties (ii) and (iii).

It is easy to verify that the algorithm computes sets Δ_0 and Δ_1 with these properties, and that, if there are sets Δ_0 and Δ_1 with these properties, that the algorithm will find them. Property (ii) is to ensure that all of Σ is determined. Property (iv) is to ensure that no $o \in \Delta_1$ is determined by the remaining objects of Δ_0 and Δ_1. Property (iii) could seemingly be relaxed to say that each $o \in \Delta_1$ is incident with at most one object in Σ', so as to allow objects which only later become constrained to be used as part of Δ_1. However there is no advantage in doing this, since objects which only

later become constrained could have been placed in Δ_0 instead. Since the algorithm first builds Δ_0 before considering constrained objects, we are free to require property (iii).

The incidence graph of the construction of Figure 3 containing two triangles in perspective from a point is shown in Figure 4, together with an augmented determining set $\Delta^+ = (\Delta_0, \Delta_1)$. The nodes of Δ_0 are shaded black, and those of Δ_1 are shaded grey. Many configurations which do not have determining sets do have augmented determining sets. We call the elements of Δ_0 *free* objects, and those of Δ_1 *constrained* objects.

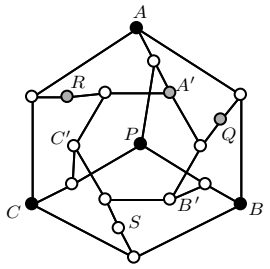

Figure 4 Incidence graph of two triangles in perspective

2.3 Proposition. *Let $\Delta^+ = (\Delta_0, \Delta_1)$ be an augmented determining set for a configuration (Σ, Π). Let $k = |\Delta_0|$ be the number of free objects and $c = |\Delta_1|$ the number of constrained objects. Then $2k + c = 2n - E$, where $n = |\Sigma|$ and $E = |\Pi|$.*
Proof. Each object $o \in \Sigma - \Delta_0 - \Delta_1$ has exactly two antecedents. There are $n-k-c$ such objects. Each constrained object has exactly one antecedent. There are c constrained objects. Therefore $E = 2(n-k-c)+c = 2n-2k-c$.

We extend the definition of *dimension* to configurations with an augmented determining set to be $\dim(\Sigma, \Pi) = k + c/2 = n - E/2$. Thus the construction of Figure 3 has dimension $3\frac{1}{2}$. Any determining set can be viewed as an augmented determining set with $\Delta_1 = \emptyset$. Thus the set of augmented determining sets for a given configuration contains the set of its determining sets. A configuration may have both augmented determining sets and determining sets. The dimension will be the same in both cases:

2.4 Corollary. *The dimension of a configuration is independent of the determining set or augmented determining set chosen.*
Proof. $n - E/2$ is a configuration invariant.

2.5 Lemma. *If $E \geq 2n - 4$ and $n \geq 4$, then (Σ, Π) does not have an augmented determining set or determining set.*
Proof. $E \geq 2n - 4$ implies that the dimension $k + c/2 \leq 2$. A dimension of 0 is impossible. A dimension of 1 is only possible if $n = 1$. A dimension of $1\frac{1}{2}$ or 2 is only possible if $n \leq 3$.

2.6 Lemma. *If every object $o \in \Sigma$ is incident with at least three other objects, then (Σ, Π) does not have an augmented determining set or determining set.*

Proof. Suppose that an augmented determining set existed. Consider the queue $Q2$. The last object on the queue would be incident with at least three objects also on $Q2$, all of which are already determined. This contradicts the definition of *determined*.

We say that a (p, q)-*configuration* is a configuration in which each point is incident with exactly p lines and each line is incident with exactly q points. An n_p-*configuration* is a (p, p)-configuration with n points and n lines. For example the configurations of Desargues' and Pappus's theorems are both (3,3)-configurations. Desargues' configuration is formed by adding the dashed line ℓ to Figure 3. Pappus's configuration is the diagram on the left in Figure 5. It is a 9_3-configuration. There are 2 other 9_3-configurations, which we call non-Pappus-1 and non-Pappus-2, also shown in Figure 5. These configurations do not have augmented determining sets, by Lemma 2.6. In the next section we shall see how we can get around this problem.

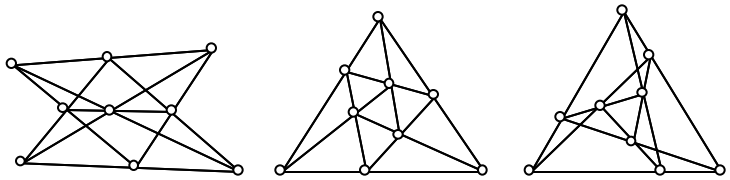

Figure 5 The Pappus, non-Pappus-1, and non-Pappus-2 configurations

Lemmas 2.5 and 2.6 say that configurations with sufficiently many incidences do not have augmenting determining sets. This is not surprising when we consider what such configurations represent. The book [1] by Bokowski and Sturmfels is devoted to the problem of finding certain coordinatizations in the plane for various configurations. Not all configurations can be drawn in the real projective plane. For example, the Fano configuration, a 7_3-configuration, cannot be coordinatized in the real projective plane. There are three 9_3-configurations, as shown in Figure 5. One of them is the Pappus configuration. Pappus's configuration is easy to draw in the real plane, since its incidences are guaranteed by a theorem. The other two are not so easy to draw. Only certain coordinatizations are possible. Similarly there are fifteen 10_3-configurations, which are listed in Bokowski and Sturmfels [1]. Of these, only Desargues' configuration can be easily drawn, since its incidences are guaranteed by a theorem. The others require special coordinatizations. Augmented determining sets allow us to select a point or line and move it arbitrarily. The remainder of the construction is thereby determined. Such arbitrary movements are simply not

possible with the configurations just mentioned. Configurations with even more incidences, for example, (3,4)-configurations or (4,4)-configurations, are much more restrictive.

When a configuration is drawn on a computer screen, we assume that the following four operations are available to construct the drawing.

M_1: Place a new point, not incident with any existing lines.
M_2: Place a new line, not incident with any existing points.
M_3: Create a new point as the intersection of two existing lines.
M_4: Create a new line as the join of two existing points.

2.7 Lemma. *Any drawing constructed with operations M_1 to M_4 has a determining set.*
Proof. The points and lines which were placed by operations M_1 and M_2 form a determining set. The points and lines created by operations M_3 and M_4 are determined objects.

Two more operations may also be available.

M_5: Place a new point, incident with exactly one existing line.
M_6: Place a new line, incident with exactly one existing point.

2.8 Lemma. *Any drawing constructed with operations M_1 to M_6 has an augmented determining set.*
Proof. The points and lines which were placed by operations M_1 and M_2 form a set Δ_0 of free objects. The points and lines created by operations M_5 and M_6 form a set Δ_1 of constrained objects. The points and lines created by operations M_3 and M_4 are determined objects.

Thus, all of the drawings that are constructed in practice do have augmented determining sets, and the algorithms described herein can be successfully used to animate them. Experience with the algorithms indicates that, if a configuration has an augmented determining set, then the backtracking algorithm will find one very quickly. Existing software that we are aware of, for example, the Cabri [3] software, allows diagrams to be animated to a limited degree. Experience with the program indicates that only points and lines which are part of the *original* determining set, i.e., that given by Lemma 2.7, can be animated. Constrained points and lines can be animated, but their movement is *restricted to the line or point constraining them*. Currently, Cabri does not allow movement of other points and lines. The algorithms described above allow these restrictions to be bypassed. In order to move a line or point, we need only find an augmented determining set containing it.

In general, (3,3)-configurations cannot be drawn by operations M_1 to M_6. This is because most of them require special coordinatizations. We will say more about this in section 4.

3. Movement

Suppose now that (Σ, Π) is a configuration and that an object $o' \in \Sigma$ is to be moved. We can initialize $Q2$ to contain o' and apply the above algorithm to find an augmented determining set $\Delta^+ = (\Delta_0, \Delta_1)$ such that $o' \in \Delta_0$, if one exists. The algorithm will save the queue $Q2$ and the antecedents $a_1(o)$ and $a_2(o)$ of each determined object o. Constrained objects will have only one antecedent, $a_1(o)$. We now scan the queue from $Q2[1]$ to $Q2[n]$, and compute the new coordinates of each object. If object o has original coordinates o, the movement applied to o is denoted by $\delta(o)$. The new coordinates are therefore $o + \delta(o)$. In the next iteration of the loop, this new position will again be denoted by o. For each object $o \in Q2$, the following three cases are possible.

- o has no antecedents. In this case $o \in \Delta_0$ is a free object. If $o = o'$, its coordinates are assigned according to whatever movement was applied to o'. If $o \neq o'$, then the coordinates of o are unchanged.
- o has two antecedents. In this case o is a determined object. Its new coordinates are given by the cross product of the new coordinates of its antecedents $a_1(o)$ and $a_2(o)$.
- o has one antecedent. In this case $o \in \Delta_1$ is a constrained object. Suppose that its antecedent $a_1(o)$ had coordinates a before it was moved, and that its movement was $\delta(a)$. If $a \cdot (a + \delta(a)) \neq 0$, then define $\delta(o) = \frac{-\delta(a) \cdot o}{a \cdot (a + \delta(a))} a$. Otherwise define $\delta(o) = a - o$. Then the new coordinates of o will constrain it to be incident with its antecedent.

3.1 Proposition. *Suppose that $o \cdot a = 0$. Let $\delta(a)$ be an arbitrary vector. If $a \cdot (a + \delta(a)) \neq 0$, then define $\delta(o) = \frac{-\delta(a) \cdot o}{a \cdot (a + \delta(a))} a$. Otherwise define $\delta(o) = a - o$. Then $(o + \delta(o)) \cdot (a + \delta(a)) = 0$.*
Proof. If $a \cdot (a + \delta(a)) = 0$, then the result is obvious. If $a \cdot (a + \delta(a)) \neq 0$, then the result can be verified by substituting for $\delta(o)$ and simplifying.

So we can animate any configuration for which an augmented determining set is available. It is also possible to express the movement as a set of equations in the homogeneous coordinates. Let $P_1, P_2, \ldots, P_n$ be the n points of a configuration, and let $\ell_1, \ell_2, \ldots, \ell_m$ be the m lines. Denote the point-line incidence matrix by A. Suppose that the coordinates of all P_i and ℓ_j are known. They satisfy the following set of equations, given in matrix form.

$$\begin{bmatrix} P_1 & & & 0 \\ & P_2 & & \\ & & \ddots & \\ 0 & & & P_n \end{bmatrix} \cdot A \cdot \begin{bmatrix} \ell_1 & & & 0 \\ & \ell_2 & & \\ & & \ddots & \\ 0 & & & \ell_m \end{bmatrix} = 0$$

If P denotes the leftmost matrix and ℓ the rightmost, this can be expressed as $P \cdot A \cdot \ell = 0$. If now P_1, say, is to be moved, the new coordinates of each object are given by solving the equations

$$(P + \delta(P)) \cdot A \cdot (\ell + \delta(\ell)) = 0$$

for the $3n$ variables $\delta(P_i)$ and the $3m$ variables $\delta(\ell_i)$, where $\delta(P_1)$ is known. This reduces to

$$\delta(P) \cdot A \cdot \ell + P \cdot A \cdot \delta(\ell) + \delta(P) \cdot A \cdot \delta(\ell) = 0.$$

The first two terms are linear, but the third one is quadratic. So we cannot expect to solve it using only linear algebra. Futhermore, the system is usually highly underdetermined because of the property that homogeneous coordinates can be freely multiplied by any constant without affecting the result. There are $3n+3m$ variables, a large number when there are only $n+m$ objects to be animated. It may be possible to solve the system iteratively, but augmented determining sets allow for a much simpler solution, as has been shown above.

4. Forcing Incidences

If the configuration of Figure 3 consisting of two triangles in perspective from a point is drawn on a computer screen, we will find that the dotted line ℓ in Figure 3 can be drawn as the join of any two of the points Q, R, and S. The third point will automatically be incident with ℓ. This is because of Desargues' theorem. Adding the line ℓ to the drawing of Figure 3 creates the Desargues configuration. However, if we add ℓ to the drawing as the join of only *two* of Q, R, S (operation M_4), the incidence graph of the configuration will not be the Desargues configuration, since one incidence will be missing. Thus we observe that the Desargues configuration with one incidence removed has a determining set. This incidence is irrelevant to the drawing, since Desargues theorem ensures that if any object in the drawing is moved, the line ℓ will always remain incident with all three of Q, R, S. A similar observation holds for the Pappus configuration.

However the same is not true of the other 9_3-configurations and 10_3-configurations. These configurations cannot be constructed so easily. We can construct the entire drawing, *but for one line*, using operations M_1 to M_6. The last line is to be incident with three points Q, R, and S. When we construct a line ℓ as the join of Q and R, we find that ℓ is not incident with S. This is because the incidences of these configurations are not guaranteed by theorems, so that they can only be coordinatized in special ways (in fact no rational coordinatizations are possible for these drawings, see Bokowski

and Sturmfels [1]). So, after constructing $\ell = RS$, we then attempt to move Q slightly so as to make it incident with ℓ. However moving Q causes ℓ to move as well. Eventually, after several iterations of this process, we have Q and ℓ aligned, at least to sight. But, if we now move any object in the diagram, Q and ℓ will drift apart. There are two observations that we make at this point.

- If we remove a single incidence from a 9_3-configuration or 10_3-configuration, then an augmented determining set exists;
- Once ℓ and Q have been aligned, we need a way to tell the program that they are incident, so that they cannot drift apart again.

We use the first observation to devise a method allowing n_3-configurations to be animated. This is presented in section 5. In this section we describe a means of implementing an algorithm for the second observation.

If a point Q and line ℓ are aligned on the screen according to sight, we may find that $Q \cdot \ell = 0.00423541$, for example; that is, they are not aligned sufficiently accurately for the computer to consider them incident. We assume that the computer requires that $|Q \cdot \ell| < \varepsilon$, where ε is a suitably small constant (maybe 10^{-6}). In order to make Q and ℓ incident, we must first adjust the diagram slightly, until $|Q \cdot \ell| < \varepsilon$, and then add the incidence $P\ell$ to the incidence graph. In order to force P and ℓ to be incident, we have used an iterative procedure. Assume that an augmented determining set Δ^+ is available for the configuration without incidence $P\ell$, such that $P \in \Delta_0$ and $\ell \notin \Delta_0 \cup \Delta_1$. The current coordinates of P are known. We are to move P by an amount $\delta_P = (x, y, z)$ to make it incident with ℓ, where x, y, and z are to be determined. The remaining objects of Δ^+ will not be moved.

Suppose that a line m of rank one is determined by points P and Q, so that $m = P \times Q$. If P and Q were moved slightly, then $\delta_m = \delta_P \times Q + P \times \delta_Q + \delta_P \times \delta_Q$. If $\delta_Q = 0$, we can write the homogeneous coordinates of δ_m as linear combinations of x, y, and z. Thus, the change in the homogeneous coordinates of any object of rank 1 is a linear combination of x, y, and z. If now $M = p \times q$ is an object of rank two, determined by lines p and q of rank ≤ 1, then $\delta_M = \delta_p \times q + p \times \delta_q + \delta_p \times \delta_q$, where the coordinates of δ_p and δ_q are linear combinations of x, y, and z. This equation can be linearized by ignoring the quadratic term. If the movements are small, this should give a good approximation. Therefore we take $\delta_M \approx \delta_p \times q + p \times \delta_q$, and the coordinates of δ_M are then also linear combinations of x, y, and z. Thus, the change in the homogeneous coordinates of any object of rank 2 is also a linear combination of x, y, and z. In general, when P is moved by an amount $\delta_P = (x, y, z)$, we can approximate the change in the coordinates of all objects as linear combinations of x, y, and z. In particular, δ_ℓ is also

a linear combination of x, y, and z. Since we want P and ℓ to be incident, we write

$$(P + \delta_P) \cdot (\ell + \delta_\ell) \approx P \cdot \ell + P \cdot \delta_\ell + \delta_P \cdot \ell = 0.$$

Each term in this expression is either a constant $(P \cdot \ell)$ or a linear combination of x, y, and z. Thus we can write this equation as

$$rx + sy + tz = -c$$

where $c = P \cdot \ell$. We now describe an algorithm that can be used to adjust the coordinates of the objects until P and ℓ are incident.

- Find an augmented determining set (Δ_0, Δ_1) such that $P \in \Delta_0$ and $\ell \notin \Delta_0 \cup \Delta_1$. P is to be moved by an amount $\delta_P = (x, y, z)$. All other objects of $\Delta_0 \cup \Delta_1$ are assigned movements $(0, 0, 0)$.
- While $|P \cdot \ell| > \varepsilon$ do
  - Calculate the δ's of all objects in terms of δ_P. These will be linear combinations of x, y, and z.
  - Evaluate the expression $P \cdot \ell + P \cdot \delta_\ell + \delta_P \cdot \ell$. Because of the simplification, this will be of the form $rx + sy + tz + c$, where r, s, t, and c are real values.
  - Solve the constraining equation $rx + sy + tz = -c$ for x, y, and z.
  - Apply the movement (x, y, z) to P and recalculate the positions of all objects in the construction.
- Add the incidence $P\ell$ to the incidence graph of the configuration.

Solving the Constraining Equation

Notice that we have a tremendous amount of freedom in choosing x, y, and z when solving $rx + sy + tz = -c$. One way would be to arbitrarily select values for x and y, say, and then solve for z. However this produces erratic movement in the diagram. Another solution is to take $(x, y, z) = (\frac{-c}{3r}, \frac{-c}{3s}, \frac{-c}{3t})$. This satisfies the constraining equation and spreads the effect of the movement more evenly across x, y, and z. Notice that $c = P \cdot \ell$ is a measure of how close P and ℓ are. As P and ℓ move closer to each other with each iteration, c becomes smaller and the applied movement becomes smaller. This helps to reduce the oscillation that can result with iteration -based algorithms.

However this method does not take into account the current position of P. Applying a relatively large movement to a small coordinate of P could drastically change the position of P. If (P_1, P_2, P_3) are the homogeneous coordinates of P, write $|P| = |P_1| + |P_2| + |P_3|$. Define the movement $(x, y, z) = (\frac{-c|P_1|}{r|P|}, \frac{-c|P_2|}{s|P|}, \frac{-c|P_3|}{t|P|})$. Then the larger coordinates of P will be changed by a greater amount than the smaller coordinates.

When these methods were tested, the last method was found to produce the *nicest* and most consistent results. For example, Figure 6 shows a 10_3-configuration which was constructed in this way. All incidences but one were constructed by the operations M_1 to M_6. The last incidence, between the dashed line and shaded point, was constructed by executing the incidence-forcing operation. We call this operation M_7:

M_7: Given a point P and a line ℓ in a configuration, force P and ℓ to be incident.

Use of the incidence-forcing operation allows n_3-configurations to be easily constructed.

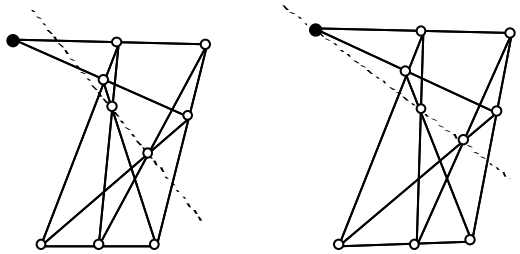

Figure 6 Result of applying the incidence-forcing algorithm.

Suppose that an incidence has been forced, thereby creating an n_3-configuration for some n. We may very likely want to adjust the drawing slightly by moving one or more of its objects. However we know that the configuration now has no augmented determining set. The next section shows how we can circumvent this problem.

5. Movement When No Determining Set Exists

Let (Σ, Π) be a configuration for which no augmented determining set exists. We describe two methods that can be used to animate the drawing. The first method is based on the incidence-forcing algorithm of the previous section. The second method is based on central collineations of the projective plane.

Suppose that an object o is to be moved. We select a point P and a line ℓ such that $P\ell \in \Pi$, and remove the incidence $P\ell$ from the incidence graph. The modified configuration will still look the same on the screen, but may now have an augmented determining set. We then find an augmented determining set such that $o, P \in \Delta_0$, and $\ell \notin \Delta_0 \cup \Delta_1$, if one exists. It is now possible to move o, but, as we do, the point P and line ℓ will start to drift apart. Therefore with each movement of o, we apply the incidence-forcing algorithm to move P and ℓ together again. When the movement of o

is complete, we restore the incidence $P\ell$ to the configuration. This method is found to work quite well in practice, although there are two difficulties which can arise.

If a construction has a determining set, it tends to have many, and they are easy to find. However it is quite possible that augmented determining sets with constraints such as $o, P \in \Delta_0$, and $\ell \notin \Delta_0 \cup \Delta_1$ do not exist. To avoid a long time spent backtracking, we have set a maximum number of iterations allowed in the search for an augmented determining set. If one is not found within the set number of iterations (for example, 30), then the algorithm gives up and refuses to allow the movement of object o. In this way we always have a linear time bound on the performance of the algorithm. If there are n objects in total, the number of steps required to animate the diagram is $O(n)$, since each animation requires running though a queue of n objects, once an augmenting determining set has been found. It is also possible to store a collection of augmented determining sets for a given configuration, such that each object is in at least one set, so that, before executing the backtracking algorithm to find a determining set, the program first checks if one is already known.

The second problem which can arise is the phenomenon of collapse. The incidence-forcing algorithm moves P and ℓ together by iteratively adjusting the positions of all objects in Σ until the process converges. Sometimes the process converges to a solution in which several points or several lines have the same coordinates. Thus on the screen they appear superimposed on each other. Once a diagram has collapsed, it remains collapsed. It can still be animated, but objects with equal coordinates will ever after have equal coordinates. In effect we have a homomorphism from the original configuration (Σ, Π) to a reduced configuration (Σ', Π'). Each $o \in \Sigma$ is mapped to some $o' \in \Sigma'$. If $P\ell \in \Pi$, then $P'\ell' \in \Pi'$. The study of configurations (Σ, Π) and their homomorphic images (Σ', Π') may be of some interest. We have also found that the particular determining set used in such cases can have a marked effect on the animation of a diagram. Whereas a diagram may collapse with a given determining set, or have erratic movement, if a different determining set is selected, the motion will be much smoother. More work needs to be done on this question.

Collineations

A *collineation* of the projective plane is an incidence-preserving mapping of the plane to itself. A *central collineation* (see Pedoe [4]) is a collineation which has a line of fixed points and a point of fixed lines. For example, Figure 7 shows a collineation α in which all points on line ℓ are fixed and all lines through point P are fixed. If we know that α maps Q onto R, that is, $Q^\alpha = R$, and we want to find S^α, this is easily done by constructing the

line QS, finding its intersection X with ℓ, and then finding the line XR and its intersection with PS. In terms of homogeneous coordinates,

$$S^\alpha = \{[(Q \times S) \times \ell] \times R\} \times (P \times S).$$

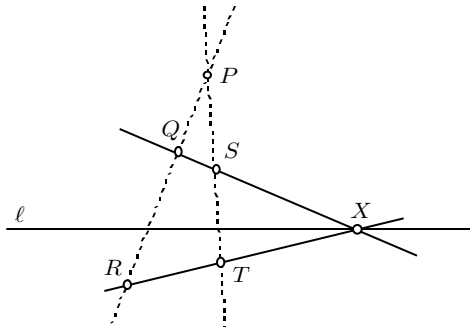

Figure 7 A central collineation determined by P and ℓ.

This is fast to compute. It is just 5 cross-products. It is easy to specify a central collineation on the computer screen. We draw a line ℓ and a point $P \notin \ell$ and tell the computer that they are to be considered as the fixed line and point of a central collineation α. When a point $Q \notin \ell$, where $Q \neq P$, is moved to some point $R \in PQ$, where $R \notin \ell$, a collineation α is specified by taking $Q^\alpha = R$.

Suppose now that two central collineations α_1 and α_2 are to be determined, with ℓ_1, P_1 and ℓ_2, P_2 fixed. A point Q in a configuration (Σ, Π) is moved to a position R. This is illustrated in Figure 8.

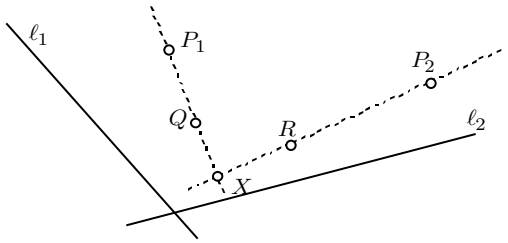

Figure 8 A pair of central collineations.

5.1 Lemma. *Given distinct lines ℓ_1, ℓ_2 and distinct points $P_1 \notin \ell_1$ and $P_2 \notin \ell_2$, let $Q \neq P_1$ and $R \neq P_2$ be points such that $Q \notin \ell_1$, $R \notin \ell_2$. Let X denote the intersection of the lines P_1Q and P_2R. If $X \notin \ell_1, \ell_2$, then there is a unique pair of central collineations α_1, α_2 such that $Q^{\alpha_1\alpha_2} = R$, where P_1, ℓ_1 correspond to α_1 and P_2, ℓ_2 correspond to α_2.*

Proof. Q can only be moved along the line P_1Q by α_1. R can only be moved along the line P_2R by α_2. Let X be the intersection of P_1Q and P_2R. If $X \notin \ell_1$, there is a unique α_1 moving Q to X. If $X \notin \ell_2$, there is a unique α_2 moving X to R. Thus $R = Q^{\alpha_1\alpha_2}$.

We can use this lemma to animate any construction for which an augmented determining set is not available. We first define ℓ_1, P_1 and ℓ_2, P_2 so that they are not part of the configuration (Σ, Π). When a point Q in the configuration is moved to a point $R = Q + \delta_Q$, we find α_1 and α_2 as in Lemma 5.1. Finding α_1 requires 5 cross products after the point X has been found. Finding α_2 requires an additional 5 cross products. We then animate the diagram by mapping the position of each object $o \in \Sigma$ to $o^{\alpha_1\alpha_2}$. Since collineations preserve incidence in the entire plane, we can be sure that all incidences of the configuration (Σ, Π) will be maintained. This movement is very quick to compute and produces a "nice" animation. It requires a constant number of steps for each movement of Q. If a line q is to be moved instead of a point Q, the dual of the lemma is used. There is a possible difficulty if it is found that $X \in \ell_1$ or $X \in \ell_2$. This does not seem to occur in practice. It can be avoided by using 3 successive collineations if it does occur.

6. Conclusion

The algorithms described in this paper have been implemented on a Sparc computer using the X-Windows system. They were found to perform very well in practice. They allow diagrams to be constructed by the following 7 operations.

M_1: Place a new point, not incident with any existing lines.
M_2: Place a new line, not incident with any existing points.
M_3: Create a new point as the intersection of two existing lines.
M_4: Create a new line as the join of two existing points.
M_5: Place a new point, incident with exactly one existing line.
M_6: Place a new line, incident with exactly one existing point.
M_7: Given a point P and a line ℓ in a configuration, force P and ℓ to be incident.

They allow an arbitrary point or line of a diagram to be selected and moved, without constraints. The entire diagram is animated so as to maintain all incidences. We finish with a list of questions.

1. Find necessary and sufficient conditions for a configuration to have an augmented determining set. Often, when moving an object P, a user wants one or more objects $o_1, o_2, \ldots$ to remain stationary in order to see the effect of moving P on other objects. Thus an augmented

determining set containing P and $o_1, o_2, \ldots$ is needed. What effect does the pre-inclusion of several objects have on the existence of an augmented determining set?
2. Is there a polynomial algorithm to determine whether a configuration has an augmented determining set?
3. Under what conditions will a configuration collapse onto a homomorphic image? What collapsed configurations are possible?
4. Investigate the effect of the choice of determining set on the animation of geometric configurations.

Acknowledgement

We would like to thank David Kelly of the Mathematics Department of the University of Manitoba for a number of helpful discussions on the topic of determining sets.

References

1. Jürgen Bokowski and Bernd Sturmfels, *Computational Synthetic Geometry*, Lecture Notes in Mathematics #1355, Springer-Verlag, 1989.
2. H.S.M. Coxeter, *The Real Projective Plane*, Cambridge University Press, Cambridge, 1961.
3. Jean-Marie Laborde and Franck Bellemain, *Cabri Geometry*, Macintosh computer software, Université Joseph Fourier, 1994.
4. Daniel Pedoe, *An Introduction to Projective Geometry*, Pergamon Press, New York, 1963.