Advances in Applied Clifford Algebras

Boosted Surfaces: Synthesis of 3D Meshes using Point Pair Generators in the Conformal Model

--Manuscript Draft--

Manuscript Number:	
Full Title:	Boosted Surfaces: Synthesis of 3D Meshes using Point Pair Generators in the Conformal Model
Article Type:	Special Issue AGACSE 2012
Keywords:	Conformal Geometric Algebra, Computer Graphics, Mesh Warping, Point Pair, Lorentz Boost, Special Conformal Transformation, Conformal Mapping, Surface Topology
Corresponding Author:	Pablo Colapinto University of California at Santa Barbara UNITED STATES
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	University of California at Santa Barbara
Corresponding Author's Secondary Institution:	
First Author:	Pablo Colapinto
First Author Secondary Information:	
Order of Authors:	Pablo Colapinto
Order of Authors Secondary Information:	
Abstract:	This paper introduces a new technique for the formulation of parametric surfaces. As previously shown by Dorst and Valkenburg, point pairs in the 5D conformal model of geometric algebra can be leveraged as generators of "simple" orbit-inducing rotors. In the current work, null point pairs are treated as surface and mesh control points which can be linearly interpolated. Here, they are used to construct continuous topological transformations. Using simple boosting rotors, some basic algorithms are proposed, including the boosting rotor which takes a circle of radius r to a line tangent to it at point p, and the boost-with-a-twist which generates a Hopf bundle. We will explore their effect when integrated in a field, and examine a few techniques for defining such rotors in homogenous coordinates: the translation of tangent vectors, the geometric product of points, and the interpolation of point pairs. Applying these rotors to points and circles provides an novel and efficient basis for creating boosted forms.

Boosted Surfaces: Synthesis of 3D Meshes using Point Pair Generators in the Conformal Model

Pablo Colapinto

Abstract. This paper introduces a new technique for the formulation of parametric surfaces. As shown by Dorst and Valkenburg in [4], point pairs in the 5D conformal model of geometric algebra can be leveraged as generators of "simple" orbit-inducing *rotors*. In the current work, null point pairs, τ, are treated as surface and mesh control points which can be linearly interpolated. Here, they are used to construct continuous topological transformations of the form $\alpha + \tau$. Using this formula for a simple *boosting* rotor, some basic algorithms are proposed, including the boosting rotor which takes a circle of radius r to a line tangent to it at point p, and the boost-with-a-twist which generates a Hopf bundle. We will explore their effect when integrated in a field, and examine a few techniques for defining such rotors in homogenous coordinates: the translation of tangent vectors, the geometric product of points, and the interpolation of point pairs. Applying these rotors to points and circles provides an novel and efficient basis for creating *boosted forms*.

Mathematics Subject Classification (2010). Primary 15A66; Secondary 53C99.

Keywords. Conformal Geometric Algebra, Computer Graphics, Mesh Warping, Point Pair, Lorentz Boost, Special Conformal Transformation, Conformal Mapping, Surface Topology.

1. Background

5-dimensional conformal geometric algebra [CGA] is a **compact** and **expressive** representation of 3-dimensional space and its admitted transformations. A thorough explanation of the conformal model can be found in [6], and a good introduction to the *rotors* (i.e. spinors) embedded in that representational model in [2]. Alternative geometries admitted by the model have been explicity investigated by physicists such as Anthony Lasenby (for instance in [5]). In the fields of computer graphics and computer vision, much attention has been paid to CGA's encapsulation of rigid body movements using dual lines to generate motors, for example in [3]. Less-explored is the more complex transformation generator: the point pair, though a rigorous mathematical treatment of logarithms and exponentials of point pairs can be found in the 2011 paper by Dorst and Valkenburg[4], in which orthogonal and commuting point pairs are shown to generate various "orbits", including knots, in 3D space. The geometric algebra graphics community has only just begun to examine the warping capabilities of these non-Euclidean operators, leaving thorough investigation of synthesis of *boosted forms* using simple point-pair rotors e^{τ} a still-unfinished task. Through consideration of various formulations of *boosting rotors* and the topology of the shapes they generate will hopefully encourage futher work in applying these methods to specific design and engineering problems. Figures below were created using `Versor`, the author's implementation of conformal geometric algebra for graphics synthesis[1].

2. Introduction: Tangent Vectors at the Origin

Boosts – also called *constant accelerations*, *transversions* or *Lorentz transformations* – are conformal operators which can straighten circles and bend lines. They can be used to model the symmetries of relativistic physics and electromagnetic fields, though here we will be considering their geometric characteristics as shape operators. In the conformal model of $\mathbb{R}^{4,1}$ developed by Li, Hestenes, and Rockwood and outlined in [6], boost operators are built as an exponential of a tangent vector. Specifically, as described by Dorst, Fontijne, and Mann in [2], we can create such an operator by combining a scalar value of 1 with a tangent *bivector generator* t, itself formed by wedging the origin n_o with a eucldiean vector $\mathbf{v}$:

$$\mathcal{B} = e^{n_o \mathbf{v}} = 1 + n_o \mathbf{v} \tag{1}$$

This spinor, or rotor, is applied to other elements of the geometric algebra using the normal 'sandwich' product:

$$x' = \mathcal{B} x \tilde{\mathcal{B}} \tag{2}$$

where x is a geometric element such as a point, circle, line, etc and $\tilde{\mathcal{B}}$ is the *reverse* of $\mathcal{B}$. In the case that x is a circle, figure1 shows what happens as we increase the length of euclidean vector $\mathbf{v}$.

In figure 1 the tangent generator lies in the same plane as the circle it is operating on. In the 5D conformal model such transformations are not limited to the 2D plane. Figure 2 shows the result of transforming along a tangent *orthogonal* to the plane of the circle.

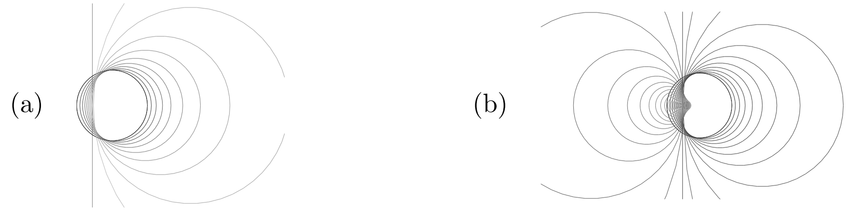


FIGURE 1. $\sigma' = \mathcal{B}\sigma\tilde{\mathcal{B}}$ with σ representing a unit circle at the origin on the e_{12} (xy) plane and $\mathcal{B} = e^{\lambda n_o \wedge e_1}$ a boost in the e_1 (x) direction. a) The circle transforms into a line when $\lambda = 1$. b) As λ increases past 1, the unit circle has turned inside-out and reverses orientation. It converges on its own center point as λ approaches infinity.

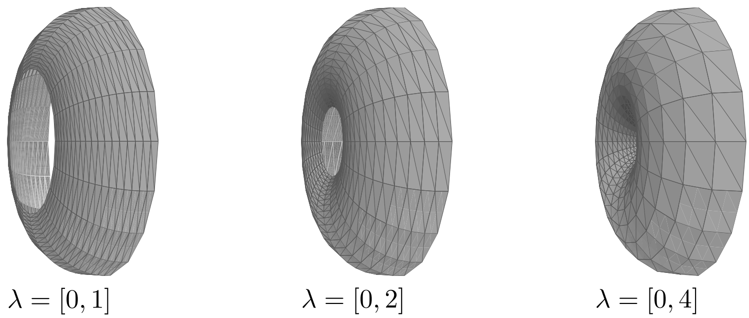


FIGURE 2. $\sigma' = \mathcal{B}\sigma\tilde{\mathcal{B}}$ with σ representing a unit circle at the origin on the e_{23} (yz) plane and $\mathcal{B} = e^{\lambda n_o e_1}$ a boost in the e_1 (x) direction orthogonal to the circle (i.e. normal to the plane of the circle). Surfaces are created by boosting σ across a range of λ values. From left to right, the range of λ values increases in each of the three surfaces.

3. Translated Tangents are Null Point Pairs

Generalized boosts which are not 'tied' to the origin are sometimes built by concatenating the above *pure* transversion rotors with translation, rotation and dilation rotors. Such concatenations are called *special conformal transformations*[1]. For instance, loxodromes[2] partially generalize the transformation by concatenating it with a translation: $\mathcal{L} = e^{n_o\mathbf{a}+\mathbf{b}n_\infty} = e^{n_o\mathbf{a}}e^{\mathbf{b}n_\infty}$. We would like to build some sort of intution about the 2-blade generators of the more general concatenations, in order to develop a single, easily differentiable and interpolatable rotor.

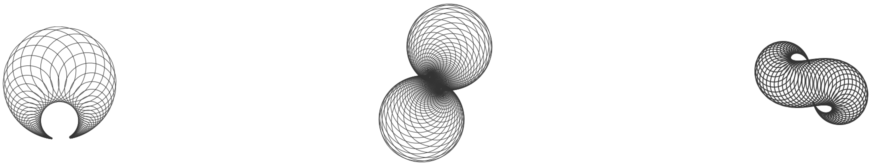


FIGURE 3. Various loxodromic transformations of the form $e^{\lambda n_o\wedge\boldsymbol{a}+\gamma\boldsymbol{b}\wedge n_\infty}$.

To generalize a boost, we can homogenize the transformation in 3D space by translating the boosting operation itself. Using translators of the form $\mathcal{T} = e^{\mathbf{v}n_\infty} = 1 - \frac{\mathbf{v}n_\infty}{2}$ we can transform elements of CGA using the sandwich product:

$$\mathcal{B}' = \mathcal{T}\mathcal{B}\tilde{\mathcal{T}} \tag{3}$$

The translation of the boost $\mathcal{B} = 1 + o\mathbf{t}$ by the translator $\mathcal{T}$ yields a scalar, a point pair, and five zero-valued quadvectors:

$\alpha + e_{12} + e_{13} + e_{23} + n_oe_1 + n_oe_2 + n_oe_3 + e_1n_\infty + e_2n_\infty + e_3n_\infty + e_{12}E + e_{13}E + e_{23}E + n_oe_{123} + e_{123}n_\infty$

where E is the Minkowski plane $n_o \wedge n_\infty$. Dropping the zero-valued terms, we are left with a scalar and a point pair: a boost spinor at position $\mathbf{v}$ in direction $\mathbf{t}$. It makes sense to see the point pair here as the bivector part of a rotor.

As mentioned by Dorst et al in [2], there is an intimate relationship between tangent vectors and point pairs: namely, a tangent vector that undergoes a translation away from the origin is a point pair with zero radius. Note that such a zero-sized element will still have a variable *weight* determined by the length of the original tangent vector. We can write

$$\tau = \mathcal{T}t\tilde{\mathcal{T}} \tag{4}$$

where τ is a point pair of zero radius, $\mathcal{T}$ is a translating spinor of the form $1 + \mathbf{v}n_\infty$, and t is a tangent vector.

---

[1]In an effort to build up the concept of a curvature operation in homogenous coordinates, this paper uses the notion of a *generalized boost* a bit more liberally than might be found in a physics paper, where *pure boosts* are considered separately from translations and rotations or dilations. Strictly speaking, these *boosted forms* may be more precisely described as *special conformal transformation forms,* and *boosting spinors* likewise *SCT spinors.* However the verb *boost* more accurately describes the mesh modelling technique that is employed, as it relates to the active techniques of *lofting* or *skinning* a surface.

[2]Loxodromes, literally "skewed paths" in ancient Greek, are S-shaped Mbius transformations achieved by stereographic projection. See figure 3.

With this formula we have a way of specifying a tangent generator at any point, not just at the origin. But we want a spinor of the form $\alpha + \tau$, where α is a scalar value – so what scalar value do we add to it? Since boosts at the origin are written as $1+t$, and translating such a multivector according to equation3 would leave the scalar value unchanged, we can assume for now that we should just assign a value of 1 to our scalar. Our formulation of a *homogenous boost* is therefore

$$\mathcal{B} = 1 + k\tau \tag{5}$$

where k is a scalar valued constant. This result conforms with Dorst and Valkenberg's rules for the exponentialization of null point pairs. For now, a simple topology we can make using equation 5 is a surface of negative curvature as depicted in figures 4 and 5.

3.1. Surfaces of Negative Curvature

Let us examine how to build one such surface. Given a circle κ we can extract the dual line axis λ on which it lies by contraction with infinity:

$$\lambda = -n_\infty \rfloor \kappa \tag{6}$$

which wedged with infinity gives us the direction bivector $\mathbf{B}n_\infty$ of the circle:

$$\mathbf{B}n_\infty = \lambda \wedge n_\infty \tag{7}$$

and by taking the dual of this we can get the direction vector orthogonal to the circle:

$$\mathbf{d}n_\infty = (\mathbf{B}n_\infty)^* \tag{8}$$

In an implementation, we can convert this direction vector to a tangent vector by copying its terms, translate the tangent to some point on the axis λ using equation 4 and form a spinor $\mathcal{K}$ using equation 5. Boosting the circle κ gradually by $\mathcal{K}$ carves out the surface of a pseudosphere – that is, a sphere with an intrinsic curvature of -1. Concatenating this transformation with a twist creates a *twisted pseudosphere* of figure 5.

3.2. A Boost with a Twist: The Hopf Fibration

As the *twisted psuedosphere* of figure 5 can be constructed by a **twist** followed by a **boost**, so can the famous *hopf fibration* be constructed by a **boost** followed by a **twist**. The fibration maps the 2-Sphere to the 3-Sphere by treating each point on the 2-Sphere as a circle. If the south pole is the base circle, then the north pole is the axis of that circle. Thus we start by noting that the fibration can be considered an spinor which takes a circle to its axis. To construct this transformation, we compose a boost which straightens the circle, followed by a twist which translates and rotates the boosted circle.

Explicitly, to map the point p with spherical coordinates θ, ϕ on a 2-sphere to a circle fiber of a 3-sphere, we start with a circle κ at the south

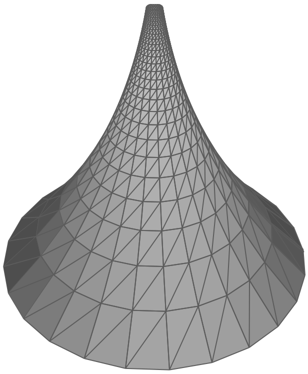


FIGURE 4. A pseudosphere built by boosting a base circle along its axis. To generate the boosting spinor $1 + k\tau$, translate the tangent orthogonal to a circle to a point along the axis and add 1. To generate the mesh, iterate transformations across a range of k (here the range is $[0, 1]$).

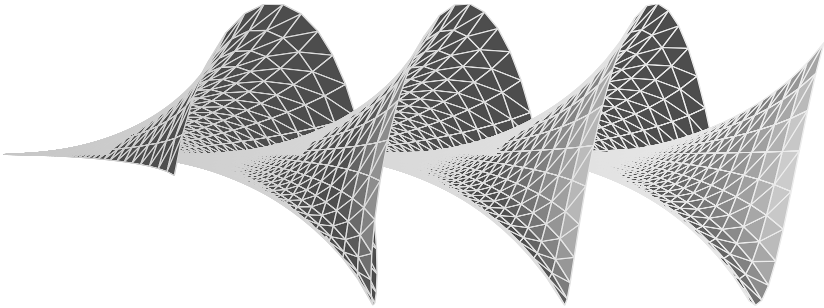


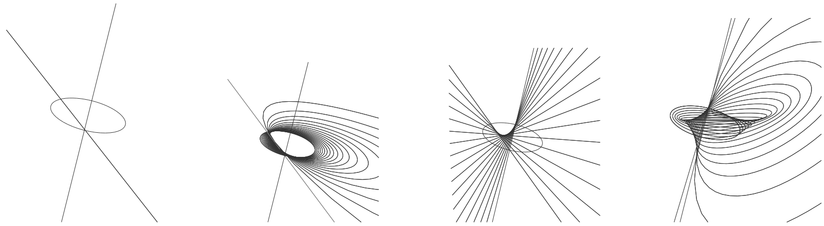
FIGURE 5. Translating the point pair generator along a dual line generator at a rate equal to the pitch of the screw, allows us to build *Dini's Surface*, sometimes referred to as a twisted pseudosphere.

pole (where $\phi = -\frac{\pi}{2}$) and a base axis $\Lambda = n_\infty \rfloor \kappa$ (where $\phi = \frac{\pi}{2}$) at the north pole. We then first define the transversion or boost $\mathcal{B}_{\theta,\phi}$ which takes the base circle to a line using equation 1:

$$\mathcal{B}_{\theta,\phi} = 1 + k_\phi n_o \boldsymbol{v}_\theta \tag{9}$$

where $k_\phi = \frac{1}{2} + \frac{\phi}{\pi}$ so that the resulting tangent term scales in the range $[0, 1]$ and where $\boldsymbol{v}_\theta$ is a unit vector rotating in the plane of the circle. Applying $\mathcal{B}$ when $\phi = \frac{\pi}{2}$ takes the circle to a line.

$$\Lambda_\theta = \mathcal{B}_{\theta,\frac{\pi}{2}}[\kappa] \tag{10}$$



a) b) c) d)

Figure 6. The initial components of a Hopf Fibration can be analyzed as a base circle and its axis. a) and b) Boosting a circle into a line. c) Twisting the line to the axis. d) Composition of the boosts followed by twists

with brackets $\mathcal{R}[x]$ used as shorthand for the full sandwich product $\mathcal{R}x\tilde{\mathcal{R}}$.

The corresponding twist $\mathcal{T}_{\theta,\phi}$ can then be generated by finding the ratio of the initial axis Λ with the line Λ_θ. For detailed analysis on how to find logarithms of motors see for instance [8].

$$\mathcal{T}_{\theta,\phi} = e^{k\phi log(\frac{\Lambda}{\Lambda_\theta})} \tag{11}$$

The full transformation rotor $\mathcal{K}$ can then be defined:

$$\mathcal{K}_{\theta,\phi} = \mathcal{T}_{\theta,\phi}\mathcal{B}_{\theta,\phi} \tag{12}$$

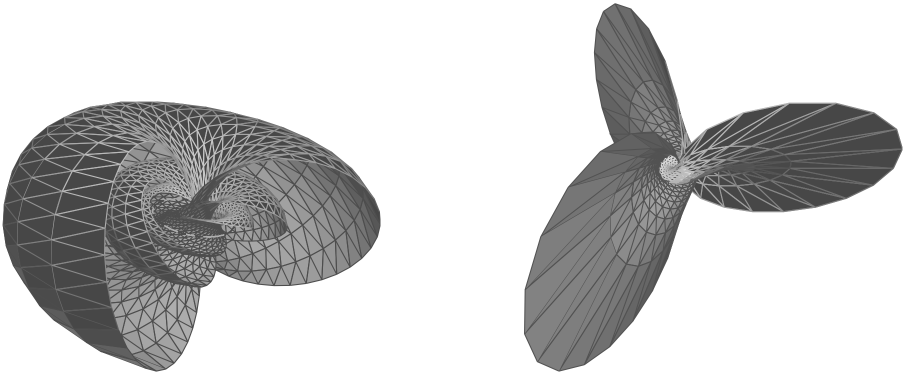


Figure 7. *Twistor* meshes representing the 3-sphere Hopf fibration generated by application of the transformation rotor $\mathcal{K}_{\theta,\phi}$ onto an originating circle.

4. Boosting Vector Fields

To better understand the transformational prowess of point pair generators let us examine their effect on a simple 2D field of points. Figure 4d. shows that a null point pair generates a simple **dipole** field. By summing such a generator with dilation and translation generators (flat points and direction vectors) one can also build **sources** and **sinks**. Sources and sinks are generated by adding to the flat point ($p \wedge n_\infty$) component of the point pair: that is, the $e_1 n_\infty$, $e_2 n_\infty$, $e_3 n_\infty$, and $n_o n_\infty$ basis blades. By adding to the direction vector only, the single dipole can be "split" into two critical points: adding in the positive direction creates a real (non-null) point pair, and adding the negative direction creates an imaginary (non-null) point pair. The resulting orbits seem to corroborate the results of Dorst and Valkenburg's work and suggests the possibility of modelling electromagnetic fields purely in terms of **critical point pair** generators of the field. It should be noted however that the equations that follow are not accurate descriptions of physical behavior, but rather can be used to generate *fields of form.*

Are other critical points possible? Perhaps by applying the bivector split methods disscused in [4], we could also represent **centers** by adding in dual line generators of rotation (though that solution is not presented here). A **focus**, then, could be created by adding in a bit of rotation around a dual line axis as well as subtracting out a bit of the flat point (to create a sink). Hyperbolic critical points, or **saddle** points, do not seem to be directly representable by point pairs.

The vector fields of figure 4 are created by a point pair τ at the center of a grid of points. Each point p is transformed to a new location by an amount inversely proportional to its squared distance. That is, for a point pair generator located at x:

$$p' = f_{cen}(e^{\lambda\tau} p e^{-\lambda\tau}) \tag{13}$$

with $\lambda = p \rfloor x$ and $f_{cen}(\sigma)$ a function that finds the center point of a dual sphere σ. This essential center-finding function eliminates changes in radii that result from the boosting of points.

4.1. Interpolation of Point Pair Generators

Now that we have a sense of how point pairs can generate fields, lets us examine how these fields might *combine* to generate surfaces. In particular, what happens when we sum more than one point pair? Given two or more point pair generators in a field, we would like to sum their impact on every position in the field. Now, there is a difference between linearly summing up all the point pairs (weighted by their squared distance to the affected field position) and then generating a single rotor from the result, as opposed to generating a (weighted) transformation rotor for each generator and summing up the displacements each one causes as is typically done in vector algebra. Though the latter is more common, here, since we are primarily interested in

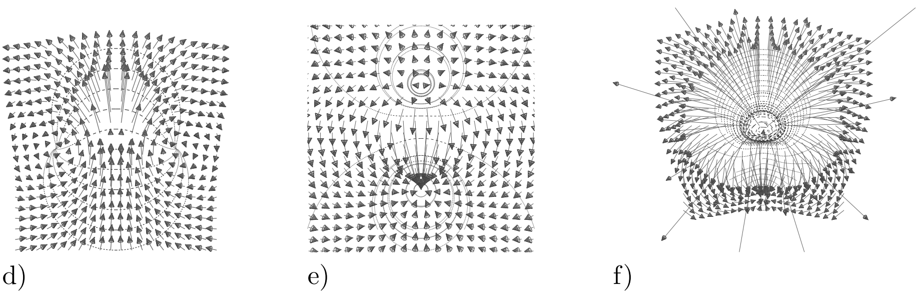


FIGURE 8. A single point pair generator τ acts on a field of points v. Equipotential contours tangent to the direction field are shown as dotted lines. a) and b) A pure sink and pure source formed by exponentialization of a flat point. c) a pure dipole formed by exponentialization of a null point pair in the y direction. d) and e) null point pair is split into imaginary and real point pairs by adding positive and negative amounts of the y direction vector. f) An extra amount of source is added to a null point pair.

creating *forms* intuitively and consistently, the first method (sum all point-pairs and then transform once) works well and provides consistent results.[3]

It is also important to note that by using this "simple" method of linear interpolation we are *not* creating suitable bivector splits as outlined in [4]. Typically in our case:

[3]In a sense, summing up point pairs *before* exponentializing has an averaging or smoothing effect – higher frequency perturbations are eliminated. Here we are interested in what forms we can create through the simplest methods available. The most "straight forward" (e.g. easiest) route is to weigh each point pair based on its distance and exponentialize that. This method creates evocative boosted forms that are no less difficult to control than using a vector-sum-of-displacements integration technique.

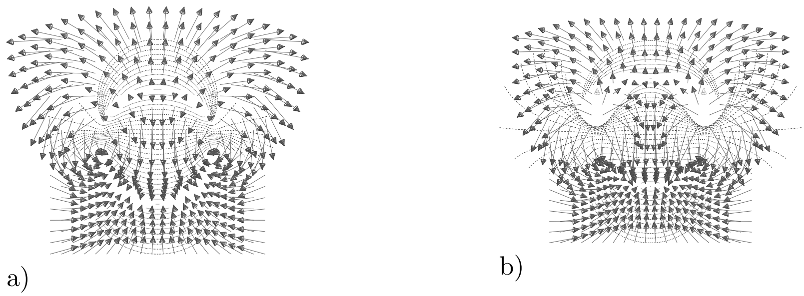


FIGURE 9. In each image, two dipole generating point pairs act on a field. Their equipotential (dotted) lines are also drawn. Two integration methods are examined: a) Summing generators first, and then creating a single rotor displacement. b) Calculating displacements for each generator and then summing the displacements. Method b) is standard in vector field topology. Method a) is the simplified approach used in this paper.

$$e^{\tau_1+\tau_2} \neq e^{\tau_1}e^{\tau_2}$$

$$\tau_1\tau_2 \neq \tau_2\tau_1$$

This is because null point pairs *only* commute if they are located at the same point in space, and we are explicity spreading them across a field. Note also that since we treat $B = \tau_1 + \tau_2$ as an affinely interpolated 2-blade we still calculate our rotor using:

$$e^{-B/2} = cosh(B/2) - sinh(B/2) \tag{14}$$

solving $sinh(X)$ using the rules for imaginary, real, and null X also defined in [4]. With these caveats and disclaimers laid bare, we proceed to examine the results of our approach on mesh warping. We start with something quite basic: we define a line contour by interpolating across a two null point pairs. The results, shown in figure 4.1, reveal an intricate relationship between the orientation generating point pairs the curvature of the resultant contour. Two null point pairs τ_1 and τ_2 are linearly interpolated to create $\tau^{'} = (1-t)\tau_1 + t\tau_2$ with t in the range $[0, 1]$. The resulting point pair, $\tau^{'}$, is used to generate a boosting rotor $\mathcal{B} = e^{-\tau^{'}/2}$. That rotor is applied to points p on the straight line connecting the two pairs.

Thus we define a *boosting field* by building a field of point pairs generators of zero size (i.e. *pure* translated tangents), and then interpolating between them using basic euler integration. 2D surface patches can then be generated through bilinear interpolation four null point pairs. A point on a surface is evaluated as

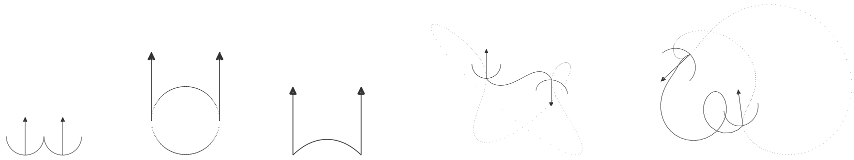


FIGURE 10. a) Point pairs τ_1 and τ_2 and their osculating circles. b) Non-null point pairs generated by affine combination of τ_1 and τ_2 . c) Transformation of the line connecting the pairs. d) and e) Different orientations. The transformed line and the interpolated pairs (shown as dots).

$$p'_{uv} = f_{cen}(\mathcal{B}_{uv} p_{uv} \tilde{\mathcal{B}_{uv}}) \tag{15}$$

where $\mathcal{B}_{uv} = e^{\tau_{uv}}$ is the boosting rotor evaluated at u, v using bilinear interpolation and $f_{cen}(\sigma)$ is our centering function which returns the center point of a dual sphere σ.

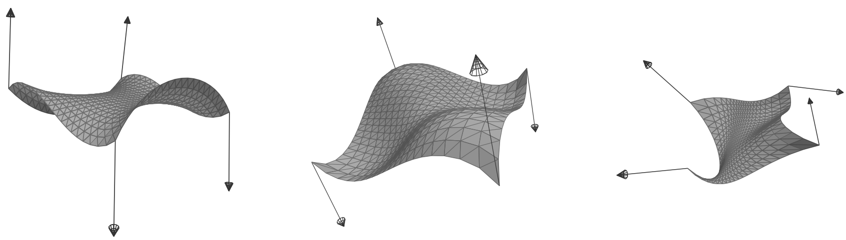


FIGURE 11. Surface patches generated by a field of boost spinors. Point pairs are first created by translating tangents to the four corners of a grid. At each vertex of grid, spinors are built by bilinear interpolation of the corner point pairs. Boosting rotors are then used to warp that point on the grid.

This approach to surface generation can be considered an extrapolation of the method using *twist fields* explicated by Wareham, Cameron and Lasenby in [7]. The key difference in methods is that by interpolating point pairs we are directly manipulating curvature at specific points over the surface, rather than interpolating between normals. Both are parametric methods offering intriguing alternatives to nurbs-based modelling.

For instance, an n-sized field of point pairs can also be used to warp a pre-existing mesh, using a sum of distance-weighted point pairs. The point pair generator to be applied to a point p is then a linear sum of n point pairs:

$$\tau_p = \sum_i^n \frac{1}{-2(p\rfloor x)} \tau_i^x \tag{16}$$

with τ_i^x a null point pair at x. Figure 4.1 shows various warpings created using this method. Each generating point pair *boosts* meshes along a tangent curve.

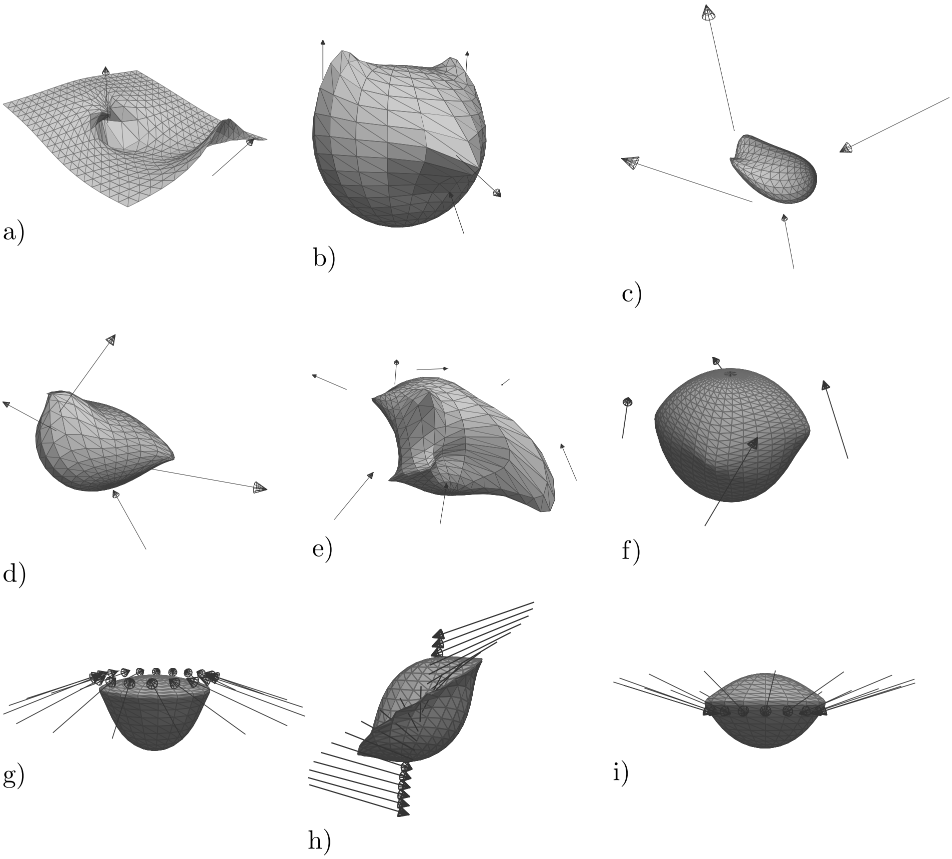


FIGURE 12. Warping of a plane (a) and a sphere (b-i) mesh by integrating multiple point pair generators. To transform each point, each generator is weighted based on the inverse of its squared distance to the point.

5. Curvature Control

From the above formulations we can begin to see the usefulness of treating boosting spinors as *curvature* operators. To solidify this role, let us investigate how to parameterize the curvature at a specific point p. The gaussian curvature of a surface at a point p is the product of the curvature of its smallest and largest osculating circles. The curvature of a circle is the inverse of its radius. Our goal is therefore to find the operation which can adjust the radius of an osculating circle while keeping the surface point fixed. To do this, we need to be able to gradually enlargen a circle into a line while fixing one of its points, and we would like a specific formula given a desired curvature. Here we identify the algorithm for continous transformation of curvature at a point p to a circle with curvature κ.

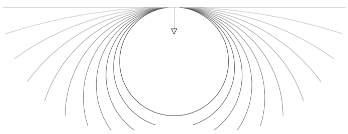


FIGURE 13. A line is 'bent' into a circle. The generator of the transformation is a tangent vector translated to a point on the line.

Given a line Λ, we can bend it into a circle of radius r and curvature $\kappa = \frac{1}{r}$ by applying a boosting spinor of the form:

$$\mathcal{B} = 1 - \frac{\kappa\tau}{2} \tag{17}$$

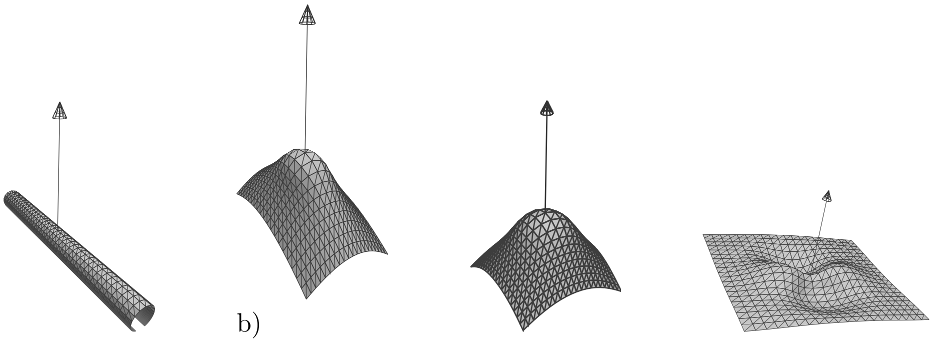
where τ is a zero-sized (null) point pair created by translating a unit tangent vector orthogonal to the line to some point p on the line, and then weighted by the negative target curvature (the inverse of the radius). By flipping the originating tangent vector around, we can use the same formula to create a boost that takes a circle with curvature κ and straightens it into a line. The $p-$translated and κ-weighted tangent τ generates a 2κ curvature operator at p.

5.1. Surface Curvature

This ability to specify curvature at a point p with an operator $\mathcal{B}$ suggests that we can build UV surface meshes by interpolating curvature control points. That is, we want to construct a surface from a set of $\kappa_n\tau_n$ operators. Consider local curvatures κ_u and κ_v in orthonormal directions u and v on a plane surface X. We generate a transforming boost in the u, v **neighborhood** of a point p on X by summing null point pairs from vector $\boldsymbol{v}$ normal to$p$ and translating in the u, v directions.

$$\mathcal{B}_{u,v}^{\kappa_u\kappa_v} = e^{\kappa_u\tau_u + \kappa_v\tau_v} \tag{18}$$

where $\tau_u = \mathcal{T}_u[n_o\boldsymbol{v}]$ and $\tau_v = \mathcal{T}_v[n_o\boldsymbol{v}]$. It is important to note that the interpolated 2-blade exponent is not always null in this case, and therefore equation 17 does not hold. The boosting rotors are therefore calculated using the rules outlined by Dorst and Valkenburg in [4] when the point pair exponent is non-null. Let us also reiterate that equation 18 does not satisfy orthogonal criteria as laid out by those authors, since the terms do not commute. For now, we can explore the shapes that such a formulation provides and see if they can be controlled intuitively. Figure 5.1 shows the results of such a formulation, which allows us to independently control u and v direction curvature at a point.



a) $\kappa_u > 0$ $\kappa_v = 0$

b) $\kappa_u > 0$ $\kappa_v > 0$ and $\neq \kappa_u$

c) $\kappa_u > 0$ $\kappa_v = \kappa_u$

d) $\kappa_u > 0$ $\kappa_v < 0$

FIGURE 14. Local curvatures defined using equation 18: a) parabolic, b) elliptic, c) umbilic and d) hyperbolic.

Integrating four such generating u, v point pairs allows us to build up a nurbs-like surface patch, as in figure 5.1. The formulation requires integrating the u curvature and v curvature separately, *before* adding them together using equation 18. That is, for each generating null tangent at the corners, we translate the tangent in the u direction along the patch and then integrate with the others using bilinear interpolation. We do the same for each tangent in the v direction. Finally we add both resultant point pairs together, and generate a boost from that sum.

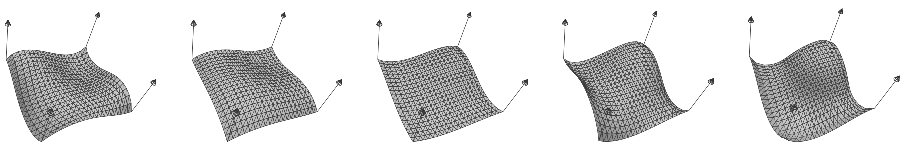


FIGURE 15. Surface patches created through independent control of curvature in the u and v directions.

6. Coda: The Geometric Product of Points

In 3-dimensional Euclidean geometric algebra it is the *ratio* of vectors and bivectors that generate continuous transformations. One might wonder, what geometric meaning can be prescribed to the ratio of points? This poses a difficulty: calculating the ratio of points is not possible since the inverse of a point is a null value.[4] We will therefore consider the geometric product of points instead. As might be expected, these products result in the same scalar + bivector formulation just described. That is,

$$p_a\tilde{p_b} = p_a p_b = s + p_a \wedge p_b \tag{19}$$

where s is a scalar and $p_a \wedge p_b$ a point pair. Thus the geometric product of points in the conformal model represent boosting operators. Since points can expand to become dual spheres, as well as have their weights modified, there is clearly much to investigate in regards to their geometric products and what sorts of transformations they admit.

Our question has now become: given two null points, what is the continuous transformation they embody? Let us first investigate the simple case of two points operating upon their dual circle: that is, two antipodes at the poles operating upon an equator. Through experimentation we find that adjustment of the weight of the null points, we can directly adjust the curvature of the transformation. Our formulation is thus:

$$K = \langle p_{aw_1} p_{bw_2} \rangle_0 + \lambda \langle p_{aw_1} p_{bw_2} \rangle_2 \tag{20}$$

where λ is a scalar range we take in the range $[-1, 1]$ and w_1 and w_2 are weights assigned to the n_o element of the points p_a and p_b.

7. Conclusion

CGA offers the graphics researcher the opportunity to discover and experiment with powerful methods for the synthesis of forms. In this paper we have explored some synthesis techniques through bending and warping using boosting rotors generated by point pairs. We have seen that such boosting operations can be used to carve out canonical manifolds such as the pseudosphere, Dini's surface, and the Hopf fibration. We have shown specific formulations for transforming curvature at an arbitrary point. Interpolating between point pair generators in a field, we were able to propose a novel design technique for the generation of organic-looking surfaces. Looking deeper, we began to establish a protocol for parametric design through curvature control. Finally, we considered primitive shape operators formed by the geometric product of weighted points. It is hoped that such purely *formal* investigations will provoke future exploration into shape operations with point pairs. For

[4]An anonymous reviewer of this paper has pointed out that the ratio of null points is not defined since points are not invertible. Computing p_b^{-1} requires one to multiply a point by its own reverse, which results in a null value. We sidestep this issue through slight abuse of concept: we will consider the geometric products of points instead of their ratio.

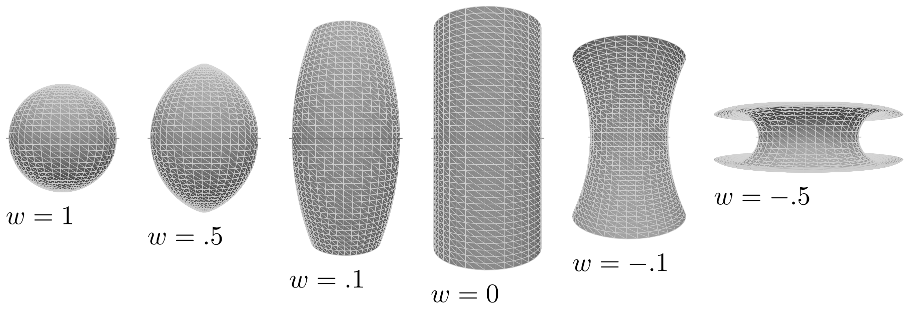


FIGURE 16. Meshes generated by continuous transformations of a circle equator (in solid black). Points at the poles (extracted by taking the dual of the circle) are weighted by assigning w to the n_o components of each. The transforming spinors are then generated by multiplying together the two points and linearly interpolating the point pair bivector part of the product from -1 to 1. As the family of spinors is applied to the circle, we can boost a surface. Note the the final surface, on the right, is nearly toroidal. At $w = -1$ the form collapses back into the equator.

instance, one could use the commutator product of a point with a point pair to establish a curvature differential operator, or evolve a network of weighted points for the construction of a new topological grammar. A formulation of *implicit* surface design using boosting operators is also desirable. The possibility of more complicated topological operations and *catastrophes* should be explored, especially those with biological significance, such as invagination and intussuceptions. Ultimately, such explorations could inspire a range of new boost-based tecniques for artistic and scientific modelling, from simulating morphogenetic dynamics to designing pneumatic structures.

References

[1] P. Colapinto, *Versor: Spatial Computing with Conformal Geometric Algebra.* Masters Thesis, University of California at Santa Barbara. Available via www/woltype.com/versor/ (2011)

[2] L. Dorst, D.Fontijne, S. Mann, *Geometric Algebra for Computer Science.* Morgan Kaufmann (2007)

[3] L. Dorst, *The Representation of Rigid Body Motions in the Conformal Model of Geometric Algebra.* In: B. Rosenhahn et al. (eds.), *Human Motion – Understanding, Modelling, Capture, and Animation,* 507–529. Springer-Verlag (2008)

[4] L. Dorst and R. Valkenburg, *Square Root and Logarithm of Rotors in 3D Conformal Geometric Algebra using Polar Decomposition.* In L. Dorst and J. Lasenby (Eds.), *Guide to Geometric Algebra in Practice* (pp. 25-46). Springer London (2011)

[5] A. Lasenby, *Recent Applications of Conformal Geometric Algebra*, in H. Li, P. J. Olver and G. Sommer (Eds.): IWMM 2004, LNCS 3519 Springer-Verlag (2005)

[6] H. Li, D. Hestenes, and A. Rockwood, *Generalized Homogeneous Coordinates for Computational Geometry*. In: G. Sommer, editor, *Geometric Computing with Clifford Algebra*, pp 25-58. Springer-Verlag (2001)

[7] R. Wareham, J. Cameron, A. Lasenby, *Applications of Conformal Geometric Algebra in Computer Vision and Graphics*. In: H. Li, P. J. Olver and G. Sommer (Eds.): 2004, Lecture Notes in Computer Science 3519, pp. 329–349. Springer-Verlag Berlin Heidelberg (2005)

[8] R. Wareham, J. Lasenby, *Rigid Body and Pose Interpolation using Geometric Algebra.* Available via http://www-sigproc.eng.cam.ac.uk/ga/static/6/63/tog_rotor_paper.pdf (2004)

Pablo Colapinto
University of California at Santa Barbara
e-mail: `pablocolapinto@gmail.com`