AGACSE 2010

Thoughts from the front line:
Current issues in real-time graphics and areas where Geometric Algebra can help

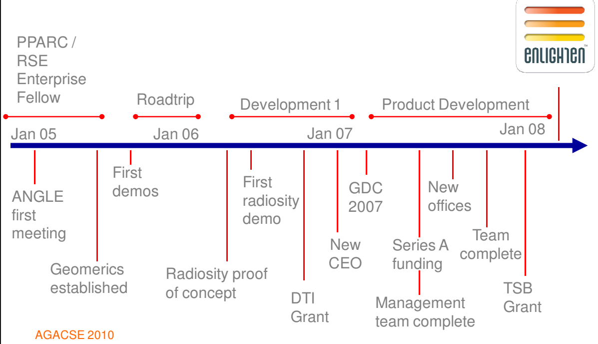
Chris Doran
Founder, Geomerics
chris.doran@geomerics.com

Introduction

- In 2005 I decided to take a break from academic research and formed Geomerics
- Looking for a new challenge
- Looking for other ways to win people over to GA

Geomerics

Geomerics Timeline



What we do

- Enlighten is a revolutionary lighting technology available for PC, Xbox 360 and PS3
- Enlighten is being used in games by EA Dice (Battlefield), CCP, Funcom, all pushing for the highest quality lighting and graphics

What we do

All images here are re-lit in real time at 30 fps, using Enlighten

AGACSE 2010

This talk

- Three areas which look interesting for future work on GA
  - Graphics
  - Discrete exterior calculus
  - Functional programming
- These are chosen not for their academic interest
- Areas where there is a real opportunity for GA to make an impact on a wider stage
- Also throwing in 3 puzzles / hobbyist topics

AGACSE 2010

Graphics

- An enormous topic, covered in depth in this conference
  - Radiosity
  - Global Illumination
  - Photon mapping
  - Ray tracing
  - Shadowing
  - Visibility
  - Ambient Occlusion
  - BRDF
  - Pre-computed radiance transfer

AGACSE 2010

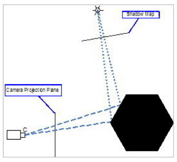
The biggest problems?

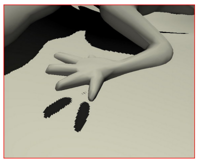
- For games, the key problems are:
- Shadowing dynamic objects from direct lights
  - Shadow maps …
- Soft shadows from area lights
  - Convolution shadow maps …
- Dynamic object radiosity
  - Dynamic ambient occlusion, screen-space techniques …

AGACSE 2010

Shadows from Direct Lights

- A solved problem to some extent:
- Create a shadow map for each light source by rendering depth information
- Use this to look up whether or not a point is in shadow
- Gives rise to jaggies, aliasing artifacts …





AGACSE 2010

Refinements

- Many ways to improve basic shadow maps
  - Deform the geometry so that the shadow map better reflects the camera orientation (paraboidal SMs)
  - Introduce a 'cascade' of shadow maps to prevent horrible blocky shadows from distant sources
- But basic problems remain:
  - High quality results using shadow maps requires high resolution maps
  - These are slow and limit the number of direct light sources that can be used in real time
  - Not obvious how to filter

AGACSE 2010

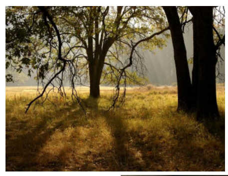
What we would like!

- A solution based on rendering from light sources that:
  - Massaged the geometry in a useful way before rendering
  - Stored more than just depth (a plane, point + line …)
  - Ideally in a form that could be low resolution and amenable to filtering
  - Implemented as a simple screen-space step (potentially where filtering came in)
- Remember:
  - Will always trade off accuracy for speed
  - A nicely blurred approximate answer often works well

AGACSE 2010

Soft Shadows

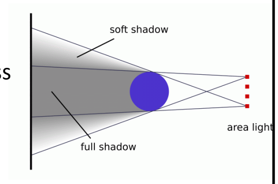
- Soft shadows are generated by area lights and are everywhere



AGACSE 2010

Soft Shadows

- This is a really hard problem!
- Can break into two aspects
  - Area lights
  - Full blown radiosity
- We have made good progress with radiosity
- But accurate area lights are unsolved for real-time graphics



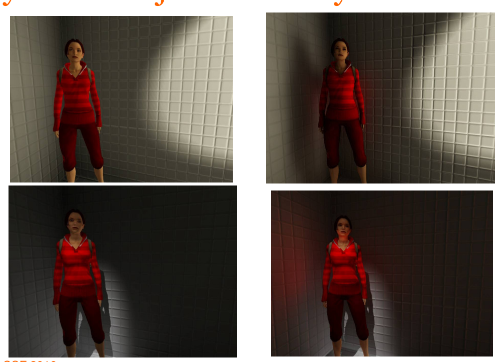
Area Lights

- The ability for an artist to dynamically place area lights with correct soft shadows would revolutionise work flow
- Any GA tricks?
  - Light sources as circles
  - Fractional / approximate visibility
  - Ability to blur simple shadows in an appropriate texture (see eg convolutions shadow maps)

Dynamic object radiosity

- In Enlighten we make a number of compromises:
- Radiosity is computed for static geometry
  - Involves an off-line pre-compute
- Light sources can move and change in real time
- Dynamic objects are lit by the radiosity
  - Appear to be rooted in their world
- But dynamic objects do not shadow the radiosity or bleed colour

Dynamic object radiosity



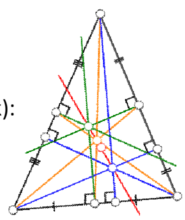
Dynamic Radiosity

- The big unsolved problem
- Need fast, approximate visibility updates
- Re-creation of form factors is less important
- Need to replace hierarchical data structures with something more malleable
- Incorporation of surface reflection properties
- Possibly screen-space type approach (caution!)
- Volume based or surface based?

Interlude 1

- Occasional frustrations with conformal GA
- Often want to drop back to affine or projective framework
- Somehow this is never easy
- Elementary pieces of geometry turn into lengthy un-inspired algebra
- Consider same basic triangle results:

Simple Triangle



- Circumcenter (green) is easy
- Centroid (orange) is (after some work):

$$G \wedge n = I(L_1 \times L_2 + L_2 \times L_3 + L_3 \times L_1)$$

- Tricky, but at least it is transparently symmetric
- Orthocentre (blue) is yet more difficult

$$H \wedge n = I \langle nL_1L_2L_3 \rangle_2$$

- Anyone got a simple proof of the Euler line?

Discrete exterior calculus

$$\int_M d\omega = \int_{\partial M} \omega.$$

- Work of Desbrun, Marsden, Hirani and others
- An attempt to develop a formal discrete theory of differential forms
  - Every continuous concept has a discrete analog
- We MUST develop a GA version of this theory
  - Otherwise the graphics community will be lost to exterior geometry for good!

Objects in DEC

- Discrete versions of each of
  - Differential forms
  - wedge product
  - Vector fields (and higher dimensions)
  - exterior derivative
  - Codifferential
  - Hodge star
  - Flat and sharp operators
  - Contraction
  - Lie derivative, Laplace – deRham operator, etc…

Foundations of DEC

- All defined in such a way that the main theorems are automatically true
- All very reminiscent of Hestenes and Sobczky's approach to the foundations of geometric calculus
- Chose your definitions carefully so that the key result is transparent

$$\oint L(dS) = \int \dot{L}(\dot{\nabla} \cdot dX)$$

Concepts in DEC

- 1-forms are numbers attached to edges
- 2-forms are numbers attached to planes
- And so on. All seems utterly obvious.
- But no useful notion of direction – the 1-form has to have the direction of the edge
- We need a notion of a vector field to discretise Maxwell equations (or anything else useful)
- At this point a dual manifold is introduced, based on either barycentric duals or centroids

Dual Manifold

- The dual manifold is the first point where things go awry
  - Vector fields look un-natural
- The wedge product is quite horrific

$$\langle \alpha^k \wedge \beta^l, \sigma^{k+l} \rangle = \frac{1}{(k+l)!} \sum_{\tau \in S_{k+l+1}} \mathrm{sign}(\tau) \frac{|\sigma^{k+l} \cap \star v_{\tau(k)}|}{|\sigma^{k+l}|} \alpha \smile \beta(\tau(\sigma^{k+l})) \,,$$

- It takes pages to prove the main results of the product
  - They should be obvious by definition
- From then on it all feels like a struggle

Hasn't this all been done?

- NO!
- Discrete exterior calculus is a recent development and actively ongoing
- Despite its difficulties it is comfortably the most complete and impressive theory we have
- With work, discrete analogs of most continuum results can be found
- We have no equivalent discrete theory within GA
  - This was not what Hestenes and Sobczyk were after

AGACSE 2010

Simple Example

- 2D vector derivative (aka the Cauchy-Riemann equations)

$$\nabla\psi = 0$$

- This is surprisingly hard to discretise
- Partly because the operator only propagates the part of the boundary data consistent with analyticity
- Can start from the Cauchy integral formula
- But then lose the ability to extend to curved surfaces
- And this is a problem of real practical significance!

AGACSE 2010

The right approach

- Some wild speculation:
  - The idea of defining scalars at points, 1-forms on lines, 2-forms on surfaces etc may not be the way to go
  - Instead, should we be defining a complete GA at discrete points?
  - Then need an operator for connecting adjacent algebras
  - This approach is more in the spirit of jet theory (see Olver: Equivalence, Invariants and Symmetry)
  - In jet theory differential equations are reduced to algebraic equations at a point, plus contact relations

AGACSE 2010

What is required

- A discrete vector manifold theory
- Based on the geometric product in the obvious way
- With a discrete vector derivative, and a discrete version of the fundamental theorem

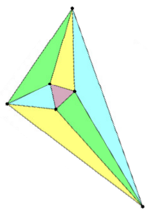
$$\oint L(dS) = \int \dot{L}(\dot{\nabla}\cdot dX)$$

- The applications for such a theory would be vast
  - EMM, elasticity, re-meshing, numerical pdes …
- This is the problem I would be focussing all efforts on!

AGACSE 2010

Interlude 2

- The Morley triangle, formed from angle tri-sectors
- Alain Connes has an algebraic proof of the result at www.alain**connes**.org/docs/**morley**.pdf
- This proof involves
  - Complex projective geometry
  - Rotations from reflections
  - Fixed points of twists
- A conformal GA version please!



Functional Programming

- Recently become interested in the functional programming language Haskell
- Will talk through its main features, and why it looks perfect for GA
- Functional languages are currently generating considerable interest:
  - Haskell, ML, ocaml …
  - Microsoft developing F#, and supporting Haskell

Haskell is a functional language

- Key objects are functions that take in arguments and return values (or functions)
- Mathematically this is simple, but far removed from modern object-oriented programming
- Means we give up on mutable objects
  - Never change a variable
  - Always create a new variable, then let garbage collector free up memory
- Focussing on functions gives compiler much better chance of parallelising code

Haskell is a ‘pure’ language

- Pure functions have no I/O side effects
- Un-used results can be discarded
- Compiler can use tricks like memoization
- Evaluations are thread-safe
  - Good for parallelisation again
- Pure functional code can have various compiler optimisations applied
- In practice, Haskell code is mostly pure with a small amount of I/O

Haskell is strongly typed

- Haskell contains a powerful type system
- Everything has a type
  - Functions map types to types, eg Int -> Int
- All code is checked for type integrity before compilation
- A lot of bugs are caught this way!
- Ties in with the concept that GA multivectors can remove ambiguity
  - Are 4 numbers are quaterion, a projective vector …
  - Tracking blades removes all ambiguity

Haskell has recursive functions

- In functional programming traditional for .. from ..to loops are replaced by other constructs
- Recursive functions are particularly useful

```
qsort [] = []
qsort (x:xs) = qsort (filter (< x) xs) ++ [x] ++ qsort (filter (>= x) xs)
```

- Use of recursion can shrink code dramatically
- Driving recursive definitions of functions is a powerful pattern matching framework
- Again, for mathematicians this is all natural!

Haskell is a higher-order language

- Functions can take functions as arguments
- Functions can return functions as results
- Under the hood, functions are curried
  - Concept due to Haskell Curry
- All functions take in one parameter, and return a function / parameter
- Great for mapping functions to lists, etc

Haskell is 'lazy'

- A defining property of Haskell is that function evaluation is lazy
- Functions are only evaluated when the result is needed elsewhere
  - Avoids unnecessary computation
  - Ensures programmes terminate where possible
  - Encourages good programming style
  - Allows for infinite lists
- Eg can define the 'infinite' list of all integers, and at a later date ask for the 10^{th} element

Haskell and GA

- This combination of properties makes Haskell uniquely suitable for GA
- Define blade and multivector data types

```
type GaBlade = (Float, GaBasis)
type GaMulti = [GaBlade]
```

- Says that a multivector is a list of blades
- Define a geometric product of blades, trivial to build up everything else
- Write code that mirrors hand-written algebra

AGACSE 2010

Laziness and GA

- Laziness is the key to Haskell's suitability

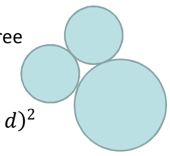
$$\langle AB \rangle_0$$

- Lazy evaluation ensures that only terms of grade zero are actually computed
- Can avoid vast amounts of hand optimisation this way
- Haskell will never be as fast as hand optimised C++ or intrinsics
- But it is far easier to write and debug, and promised much on multicore devices

AGACSE 2010

One final problem

- A fun problem from Martin Gardner's mathematical recreations
- Given three kissing circles:
  - Can always find two circles to kiss all three
- Inverse radii satisfy



$$a^2 + b^2 + c^2 + d^2 = \frac{1}{2}(a + b + c + d)^2$$

- A neat problem in conformal GA!

AGACSE 2010

Conclusions

- Many interesting open problems to explore with GA
- Opportunities to make a real difference in areas that will get GA widely noticed
  - Graphics, discrete theory, functional programming
- Plenty of drive from industry in setting the problem space, if people are interested
- And please come and talk to me if you make serious progress in any of these areas!

AGACSE 2010

Contact Details

Chris Doran
Geomerics Ltd
City House
Hills Road
Cambridge

chris.doran@geomerics.com
c.doran@mrao.cam.ac.uk
www.geomerics.com

Geomerics