Perspex Machine Tutorial

SPIE 2005

Dr. James A.D.W. Anderson
Computer Science
The University of Reading
England

Paradigm Shift

- The perspex machine is a super-Turing machine that arose from the unification of projective geometry with the Turing machine[5,6]. The word "perspex" is in[2].
- The perspex machine is intended to bring about a paradigm shift in the theory, practice, and philosophical appreciation of computation.
- It is difficult for people with established views to accept a paradigm shift. By contrast, my students accept it easily and become perspex programmers within a few weeks of reading the papers.
- This tutorial aims to give a very simple introduction to the perspex machine so that everyone can appreciate it.

Agenda - Perspex and Turing

The perspex machine is *not just* a computer. It is a super set of the Turing machine that models the universe. The perspex machine:

- Can always halt.
- Need not have incomputable numbers.
- Can always be deterministic.

Agenda - Perspex and Turing

Nonetheless the perspex machine can *emulate* the Turing machine exactly, and *simulate* it approximately because:

- The Turing machine is a degenerate perspex machine and the universal perspex machine can emulate any perspex machine.
- Physical computers can simulate the Turing machine and anything physical can be simulated by the perspex machine.

Agenda - Practical Applications

The perspex machine cannot be *emulated* by a Turing machine, but it can be *simulated* by a Turing machine.

- The simulation of super-Turing perspex properties by a Turing machine is necessarily poor!
- The perspex machine has practical utility as a guide to thinking about computation.

  How easy would it be for you to implement a compiler with the following properties?

Agenda - Practical Applications

Any finite or infinite Turing program can be compiled on a perspex machine so that:

- It is resistant to the erasure of symbols and makes partial recoveries as it degrades gracefully to total non-functioning[7].
- It operates if started between symbols and degrades gracefully, with partial recovery, as its performance declines, possibly, to zero[7].
- It can be approximated by one instruction. The approximation improves, with relapses, as instructions are added. When all of the instructions are present the approximation is exact. See[7].

Agenda - Philosophy

- Visualisation is a more competent form of reasoning than language or any symbolic system[6].
- Perspex Thesis: *The perspex machine can describe the physical universe, including mind, to arbitrary accuracy and, conversely, the physical universe, including mind, instantiates the perspex machine*[6].
- The perspex machine provides one solution to the mind-body problem by unifying computation, and hence mind, with geometrical, and hence physical, bodies[6].
- Perspex formulas have been given for consciousness and free will[1,3,6].

Agenda - Philosophy

- If there is anything that cannot be discussed in terms of perspexes then the perspex thesis fails.
- Therefore the perspex thesis is supremely falsifiable, and is, thereby, a scientific thesis.
- The perspex machine resolves some theological problems and problems with time travel.
- Conversely, if it could be shown scientifically that the perspex machine does not apply to theological questions, or to time travel, that demonstration would, itself, be a scientific limitation to the properties of God and time.

Perspex and Turing

How Numbers are Defined

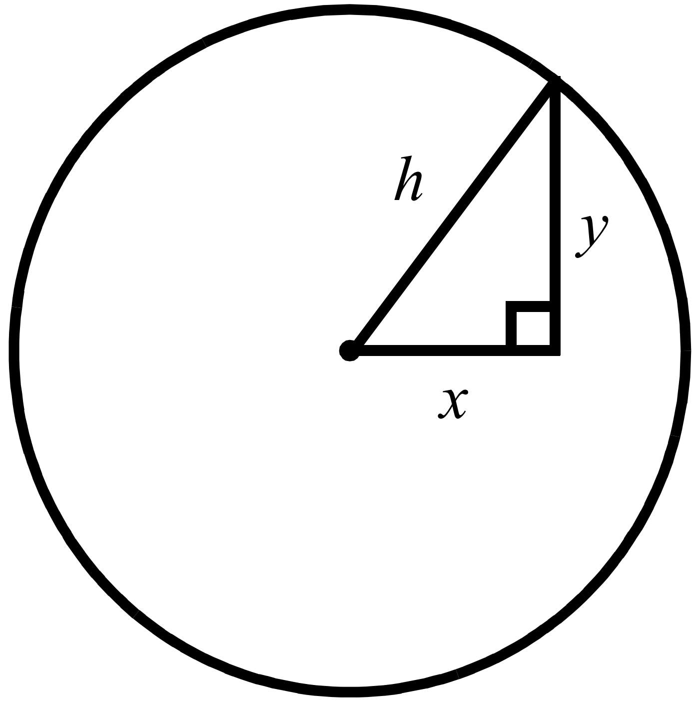
- Traditionally, numbers are defined as the solution set of an equation. For example, the equation $x^2 = -1$ led to the development of the complex numbers.

- Traditionally, numbers are defined via tables, say, for addition and multiplication.

- Traditionally, numbers are defined by axioms that describe the tables and the solution set exactly.

Rational Trigonometry

Suppose we want to perform rational trigonometry where the sides x, y, and $h = 1$ of a right angled triangle, as shown, are rational numbers.



Rational Trigonometry

Then, as was known to Euclid, the length of the sides of the triangle is given by a simultaneous diophantine equation in integer parameters n and d. See[4].

Let $x = p, y = q, h = r/r$

then $p = d^2 - n^2, q = 2dn, r = d^2 + n^2$

and the solution set is given by the Cartesian product $n \times d$.

Rational Trigonometry

However, the Cartesian product defines solutions infinitely often.

An equivalence class with each solution defined exactly once is given by:

$$\frac{n}{d} \in Q \cup \left\{ \frac{0}{0}, \frac{1}{0} \right\}.$$

Therefore the fractions nullity, $\Phi = \frac{0}{0}$, and infinity, $\infty = \frac{1}{0}$, are numbers.

Transrational Equivalence Table

- Here[4] $x, y \in Z^+$.

$\equiv$	$\frac{x}{y}$	$\frac{-x}{y}$	$\frac{x}{-y}$	$\frac{-x}{-y}$	$\frac{0}{y}$	$\frac{0}{-y}$	$\frac{x}{0}$	$\frac{-x}{0}$	$\frac{0}{0}$
x/y	T	$F_{1,2}$	$F_{1,2}$	T	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	F_2
$(-x)/y$	$F_{1,2}$	T	T	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	F_2
$x/(-y)$	$F_{1,2}$	T	T	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	F_2
$(-x)/(-y)$	T	$F_{1,2}$	$F_{1,2}$	T	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	F_2
$0/y$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	T	T	$F_{1,2}$	$F_{1,2}$	F_2
$0/(-y)$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	T	T	$F_{1,2}$	$F_{1,2}$	F_2
$x/0$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	T	T	F_2
$(-x)/0$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	$F_{1,2}$	T	T	F_2
$0/0$	F_2	F_2	F_2	F_2	F_2	F_2	F_2	F_2	T

Consistency

- Tables for addition and multiplication are readily obtained from the formulas in[4].

- These tables contain rational arithmetic as a proper subset so transrational arithmetic contains rational arithmetic as a proper subset.

- The strictly transrational parts of the tables are consistent with the rational parts because they are distinct from them.

- For example, $\infty^{-1} = 0$ and $0^{-1} = \infty$ are consistent with rational arithmetic, because ∞ is not accessible to rational arithmetic. Note: $\infty\infty^{-1} = \Phi \neq 0, 1$.

Consistency

- Infinity is non-distributive because, with $x_1, x_2 \in Q$ such that $x_1 + x_2 = x_3$ we have $\infty(x_1 + x_2) = \infty x_3 = \infty$ but $(\infty x_1) + (\infty x_2) = \infty + \infty = \Phi$.
- Non-distributivity over infinity in transrational arithmetic is consistent with the non-distributivity of terms in an infinite series of rational numbers.

Consistency

- Nullity is trivially distinct from the rational numbers because $\Phi \neq 0$ yet $x + \Phi = \Phi$ and $x\Phi = \Phi$ for all rational, infinity, and nullity x.

- Note: $\Phi\Phi^{-1} = \Phi \neq 0,\ 1$.

Consistency

Existing axiomatic proofs of the inconsistency of division by zero in arithmetic are irrelevant because:

- Their axioms do not describe the transrational tables.
- Division by zero is consistent with trigonometry.

Transrational Numbers

It follows that:

- $\infty > x, x \in Q$. That is, infinity lies at the positive extreme of the rational numbers[4].
- $\Phi \neq x, x \in Q \cup \{\infty\}$. That is, nullity lies outside the rational numbers and infinity[4].
- Nullity, Φ, and infinity, ∞, are called *strictly transrational numbers* to distinguish them from the *rational numbers* that together make up the set of *transrational numbers*.

Transreal Numbers

Replacing $x, y \in Z^{+}$ with $x, y \in R^{+}$ in the previous derivations makes Φ and ∞ strictly transreal numbers[6] with all of the given properties holding over $R \cup \{\Phi, \infty\}$ not just over $Q \cup \{\Phi, \infty\}$. In particular:

- Transreal arithmetic contains real arithmetic as a proper subset.
- Transreal arithmetic is consistent with real arithmetic.
- Infinity lies at the positive extreme of the number line.
- Nullity lies off the number line augmented with infinity.

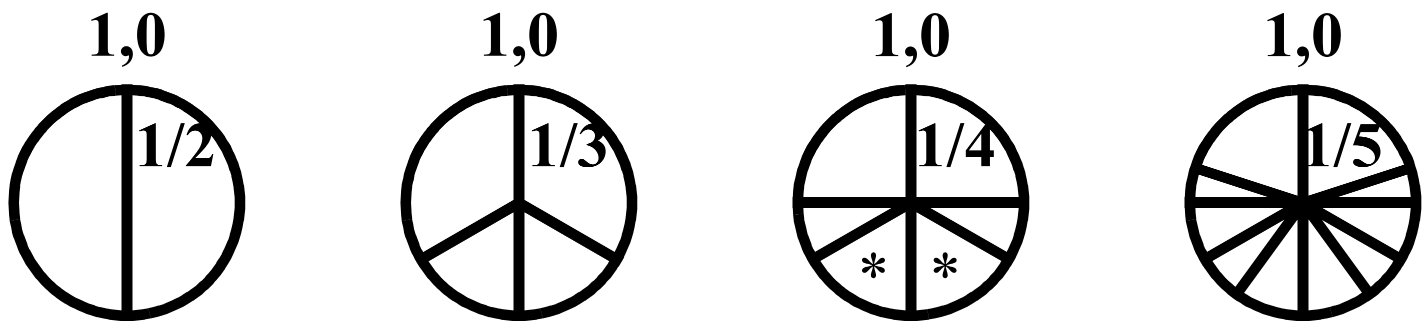
Totality

- Transreal arithmetic is total.
- Totality allows the perspex machine to be universally deterministic, whereas the Turing machine can fail on non-deterministic programs.
- The perspex machine can, nonetheless, give an exact emulation of the Turing machine by using deterministic perspex states to model non-deterministic states in a Turing machine.
- The perspex machine has no need of Turing's *oracle* or *choice machine,* because the perspex machine is total.

Walnut Cake Theorem

Suppose that a quantity is bounded above and below by rational bounds, or by an integral number of real bounds that equally subdivide the space. Then increasing the number of subdivisions does *not* necessarily tighten the bounds on a true value and can lead to a less accurate approximation at the newly introduced bounds[6].

Segments where no improvement is found are shown with an asterisk *.



Walnut Cake Theorem

In general, it can be shown that if all measurements with bounds a multiple of $\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots, \frac{1}{n}$ have been taken then a tightening of the bounds is *not* guaranteed before a single measurement at a precision of $\frac{1}{n^2}$.

This result can be obtained by the sum of an arithmetic series[6] or by the Chinese Remainder Theorem.

- Hence rational, or a finite number of equally spaced, bounds generally improve the accuracy, at the bounds, of an approximation to a true value *non*-monotonically.

Walnut Cake Theorem

- Calculus deals with the case where bounds are replaced by tighter bounds. Hence the tightness of the bounds, and the accuracy of an approximation to a true value at a bound, increases absolutely to a limit.

- Calculus assumes that measurements are made to infinite, i.e. real numbered, precision.

- By contrast, the Walnut Cake Theorem deals with the case where a lattice is replaced by a finer lattice. Here the tightness of the measured bounds, and the accuracy of an approximation to the true value at a bound, can be, and generally is, less tight and less accurate than on the sum of all the preceding lattices.

Walnut Cake Theorem

- The Walnut Cake Theorem applies where point-wise measures of accuracy can be made to a limited, but increasing, precision – not to an infinite precision.
- This is commonly the case in the real world as the precision of practical measurements increases over time with technological progress.
- Presumably, the Walnut Cake Theorem applies to animal brains where a synapse moves to a new position by growth and the effectiveness (precision) of the synapse increases over time as a collection of neurons moves into optimal positions.

Walnut Cake Theorem

The Walnut Cake Theorem gives rise to:

- Repeated partial recovery during the progressive erasure of a perspex program[7].
- Repeated relapses during the progressive increase in the number of terms in an approximation to a perspex program. See[7].
- This non-monotonic performance is a consequence of the geometry of the perspex machine, not of the text of a Turing program that is executed on the perspex machine[7].

Perspex as Matrix

- A perspex is a 4×4 matrix of transreal numbers.
- It is useful to describe the perspex by the column vectors x, y, z, t.

$$\begin{bmatrix} x_1 & y_1 & z_1 & t_1 \\ x_2 & y_2 & z_2 & t_2 \\ x_3 & y_3 & z_3 & t_3 \\ x_4 & y_4 & z_4 & t_4 \end{bmatrix}$$

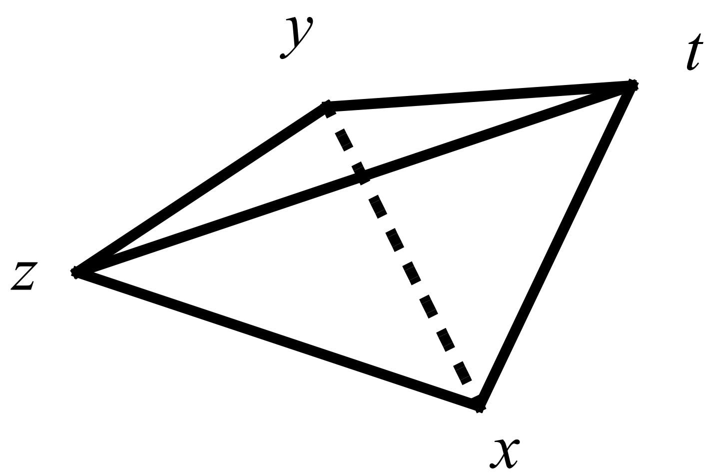
Perspex as Transformation

The perspex can describe:

- Linear transformations of transreal Cartesian co-ordinates.
- Perspective transformations of transreal homogeneous co-ordinates.

Perspex as Tetrahedron

The column vectors of a perspex can describe the vertices of a tetrahedron.



Perspex as Tetrahedron

- In the classical homogeneous model of perspective space a vertex, v, is “at infinity” if $v_4 = 0$.
- In the Cartesian model of Euclidean space a vertex, v, is “at infinity” if any co-ordinate $v_i = \infty$.
- In both models a vertex, v, is “at nullity” if $v_i = \Phi$.
- In both models the “point at nullity” has co-ordinates (Φ, Φ, Φ, Φ).
- The point $(0, 0, 0, 0)$ is punctured from the classical homogeneous model of perspective space and can be considered a synonym for (Φ, Φ, Φ, Φ). See[2].

Perspex as Instruction

A perspex can be the instruction $\vec{x}\vec{y} \rightarrow \vec{z}; \mathrm{jump}(\vec{z}_{11}, t)$.

- The superscript arrow denotes indirection. For example, x is a column vector denoting a position in 4D space, but $\vec{x}$ is the contents of that point.
- Every point in perspex space contains a perspex that defaults to the halting perspex H.
- A perspex machine is programmed by initialising space with, typically, non-H perspexes.
- A perspex machine is started by starting execution at some point or points in space.

Perspex as Instruction

Perspex instruction: $\vec{x}\vec{y} \rightarrow \vec{z};\ \mathrm{jump}(\vec{z}_{11}, t)$.

- $\vec{x}\vec{y}$ is the product of perspexes, i.e. the product of the matrices stored at the points x, y.
- In the past[5,6] the product was reduced to a standard form to implement the equivalence class used in projective geometry ($kP \equiv P,\ k \neq 0,\ k \in R$), but this makes it cumbersome for a perspex machine to copy a program that contains non-standard perspexes (it can be done using indirection) and to effect a computed jump backwards in time, t, (which can be done by using jump tables, supplied with the initial program, that contain non-standard perspexes).

Perspex as Instruction

- It is simpler to use the instruction:

  $\vec{x}\vec{y} + \text{continuum}(\vec{z}) \rightarrow \vec{z};\ \text{jump}(\vec{z}_{11}, t)$

  where $\text{continuum}(\vec{z})$ returns $\vec{z}$ if $\vec{z}$ is not the nullity matrix, but returns the zero matrix otherwise.

- This leads to an augmented Euclidean geometry that can model projective geometry.

- Thus, the Turing results stand regardless of whether the augmented Euclidean geometry or perspective geometry is used.

Perspex as Instruction

Perspex instruction:

$\vec{x}\vec{y} + \text{continuum}(\vec{z}) \rightarrow \vec{z};\ \text{jump}(\vec{z}_{11}, t)$.

- The binary arrow denotes assignment so $\vec{x}\vec{y} + \text{continuum}(\vec{z}) \rightarrow \vec{z}$ writes the product, $\vec{x}\vec{y}$, plus the translation term, $\text{continuum}(\vec{z})$, into the contents of the point z.
- The jump part of the instruction examines the top-left element of the product, $\vec{z}_{11}$, and passes control via a relative jump, j, to some point in space.

Perspex as Instruction

Perspex instruction:

$\vec{x}\vec{y} + \mathrm{continuum}(\vec{z}) \rightarrow \vec{z};\ \mathrm{jump}(\vec{z}_{11}, t).$

- If $\vec{z}_{11} < 0$ then $j_1 = t_1$, otherwise $j_1 = 0$.
- If $\vec{z}_{11} = 0$ then $j_2 = t_2$, otherwise $j_2 = 0$.
- If $\vec{z}_{11} > 0$ then $j_3 = t_3$, otherwise $j_3 = 0$.
- $j_4 = t_4$ unconditionally.

Perspex as Instruction

- The perspex instruction $\vec{x}\vec{y} + \text{continuum}(\vec{z}) \rightarrow \vec{z}; \text{jump}(\vec{z}_{11}, t)$ is total. That is, it can be executed for any transreal perspex with column vectors x, y, z, t.
- By contrast the Turing machine is partial. It cannot execute a non-deterministic program where the current machine state and symbol on the tape instructs a transition to more than one machine state or read/write/move instruction.
- Turing introduced the *choice machine* instructed by an external *oracle* to handle this non-deterministic case.
- These are not needed by the perspex machine.

Essential Data Representations

- If a Turing machine simulates a perspex machine the simulation is deterministic.
- Such a simulation can emulate a universal Turing machine.
- This is to say that a data representation – the perspex machine – gives the Turing machine access to computations, here deterministic ones, that it cannot perform as a raw Turing machine. (The example of interpolating between Turing symbols is given later.)
- In other words, some data representations are essential to delivering the computations a Turing machine is supposed to be able to deliver.

Non-Determinism

- Non-determinism is a weakness of the Turing machine.
- It is better to remove this weakness, say, by using a perspex machine instead of the Turing machine.

The Continuum

- The Turing machine is defined to operate on a finite number of symbols, but it potentially operates on an asymptotic infinitude of symbols bounded above by the cardinality of the integers: $|Z| = \aleph_0$.
- The perspex machine potentially operates on an existential infinitude of perspexes with the cardinality of the continuum, i.e. the cardinality of the real number line: $|R| = \aleph_1 = 2^{\aleph_0}$.
- Thus, the perspex machine has exactly enough perspexes to compute the power set of the integers, because this set has $2^{\aleph_0} = \aleph_1$ members.

The Continuum

- Consequently, the perspex machine can compute any real number, via the Dedekind cut. Hence all real numbers are perspex computable.

  However, this property is *not* technologically accessible.

- In other words, the perspex machine is super-Turing because all real numbers are computable.

  But, this property is *not* technologically accessible.

The Continuum

The perspex machine has more than enough perspexes to solve the halting problem, making it super-Turing.

- Keep a trace of the execution of a perspex simulation of a Turing machine. Write the first executed operation at a distance 0 parallel to a time line, the second at $1/2$, the third at $3/4$, and so on.

- When control has passed from distance 0 to beyond 1 in the perspex machine, after one unit of time, the simulation is halted and is examined to see whether or not the Turing machine was in a halt state. If it was, the program is Turing computable, but if it was not, the program is Turing incomputable. This solves the halting problem, but is *not* technologically accessible.

The Continuum

Mapping an infinitely long Turing program, or an infinitely long trace of a Turing program, onto a line of unit length has important consequences:

- Any reconstructible filter that operates on the unit line can analyse and synthesise the entire, infinite, Turing program. (A finite example is given later.)
- We might take practical advantage of this by defining an infinite program analytically on the unit line and by operating on reconstructions to a finite, not infinite, precision. This ought to encompass, for example, the whole of classical, numerical analysis.

Symbols

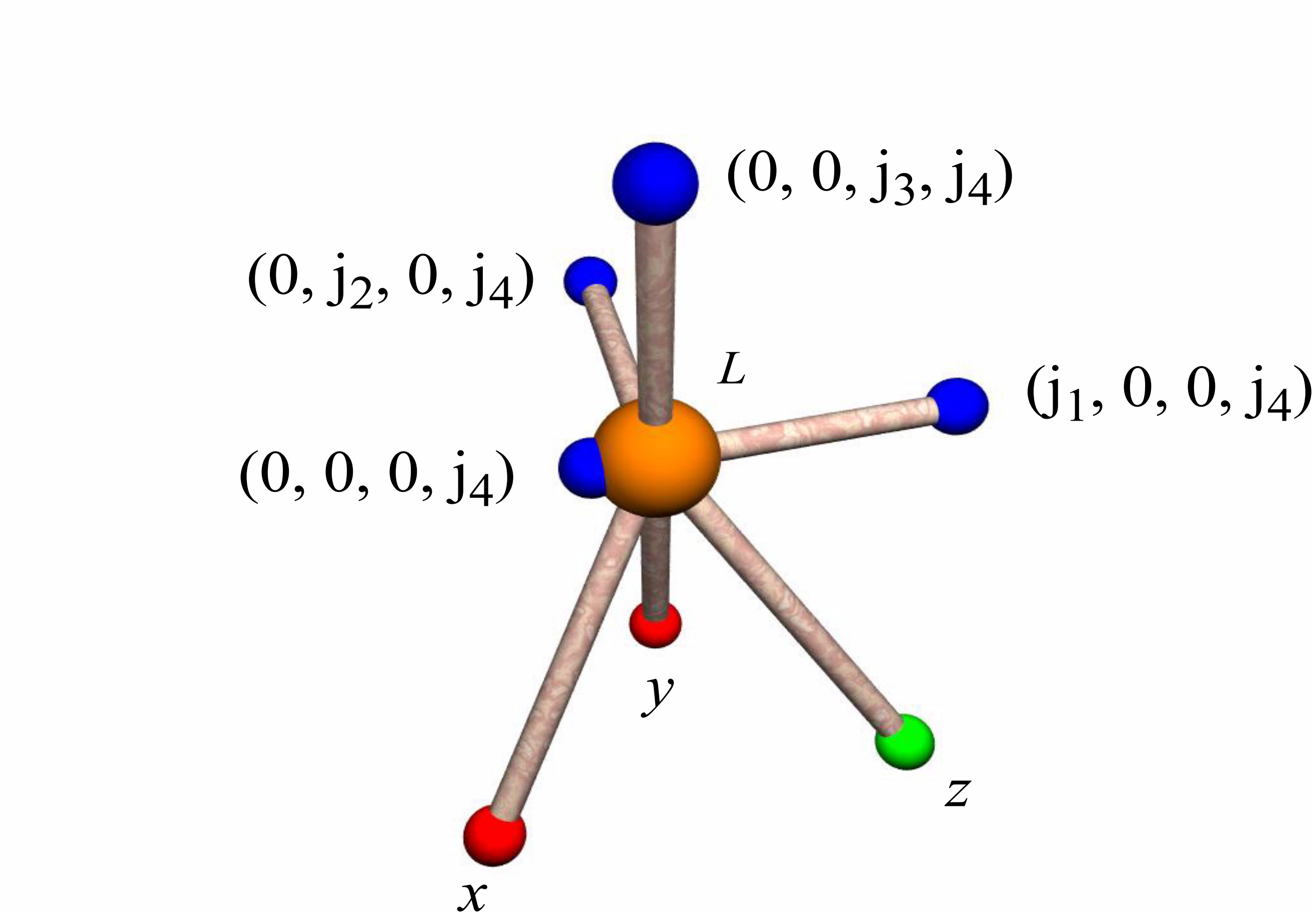
- A Turing machine operates on a finite vocabulary of discrete symbols. Hence it is susceptible to Gödel numbering and a sufficiently complex Turing machine finds that some Gödel numbers are incomputable.

- A perspex machine operates on the infinite, $\aleph_1$, vocabulary of all transreal perspexes. Hence it is immune from Gödel numbering and no incomputable Gödel numbers need exist.

- Of course, if a perspex machine emulates a Turing machine, the emulation has a Gödel numbering and may have incomputable numbers.

Perspex as Neuron

The perspex instruction can be implemented as an artificial neuron stored at a location *L* in space.



Perspex as Neuron

A network of perspex neurons has all of the properties of the perspex machine. In particular:

- It can restrict operation to discrete symbols, i.e. it can operate linguistically or logically, and be susceptible to all of the restrictions of a Turing machine.
- Or it can operate in a geometrical continuum, i.e. by visualisation, and be immune from all linguistic, logical, and Turing restrictions.
- It is an open question of physics whether biological neurons are discrete or have access to a continuum.

Super-Turing

- The perspex machine is total, but the Turing machine is partial. We will make great use of this in practical applications where it is important to know that the perspex machine does not have any error states.
- The perspex machine can solve the halting problem. Unfortunately, we know of no practical way to exploit this property.
- The perspex machine can compute all real numbers. Unfortunately, we know of no practical way to exploit this property.

Super-Turing

- The perspex machine is not susceptible to Gödel numbering so no incomputable Gödel numbers exist, unless the perspex machine emulates a Turing machine. Unfortunately, we know of no practical way to exploit this immunity.

- The perspex machine can approximate an infinite, and therefore Turing incomputable, program to arbitrary precision. It might be possible to develop practical applications of finite approximations to infinite programs.

- It is certainly possible to develop practical applications of finite approximations to finite programs. See[7].

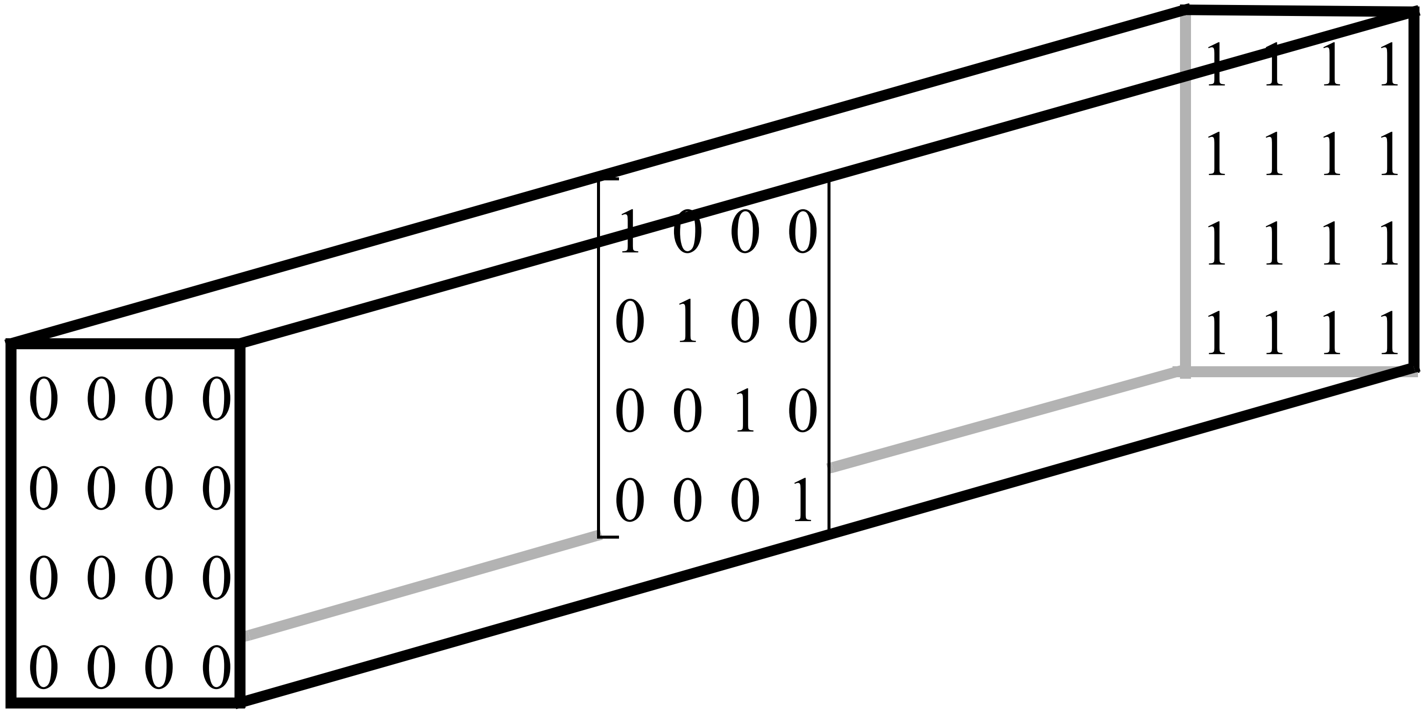
Practical Applications

Access Column

- The practical utility of the perspex machine would be increased if there were a well formed way to access individual elements of the perspex.
- The access column[6] is one way to do this. A column of $2^{16} = 65,536$ perspexes is created with a built-in side effect that allows every combination of elements to be read or written depending on their depth into the column.
- (Another way is to use the augmented Euclidean geometry described earlier.)

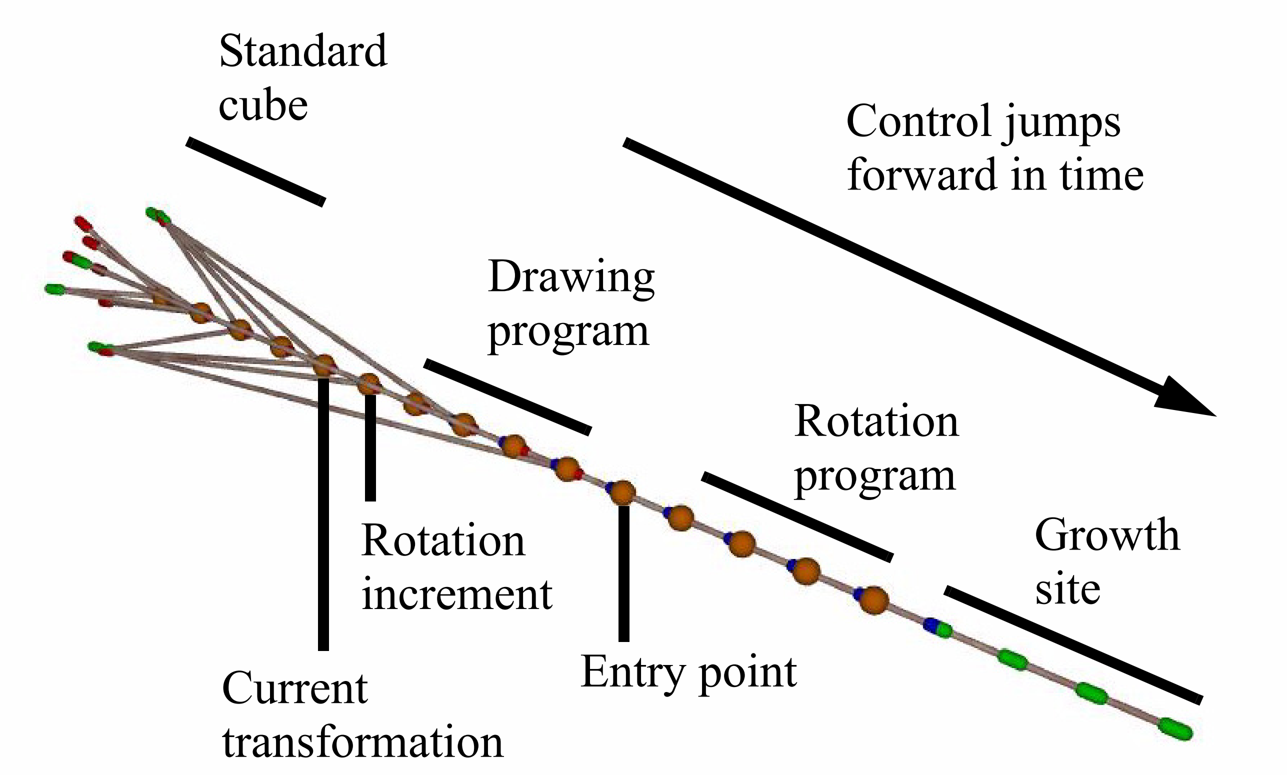
Access Column

- Here zero denotes an element to be read from/written to the first perspex in the column.
- Here unity denotes an element to be read from/written to the last perspex in the column.



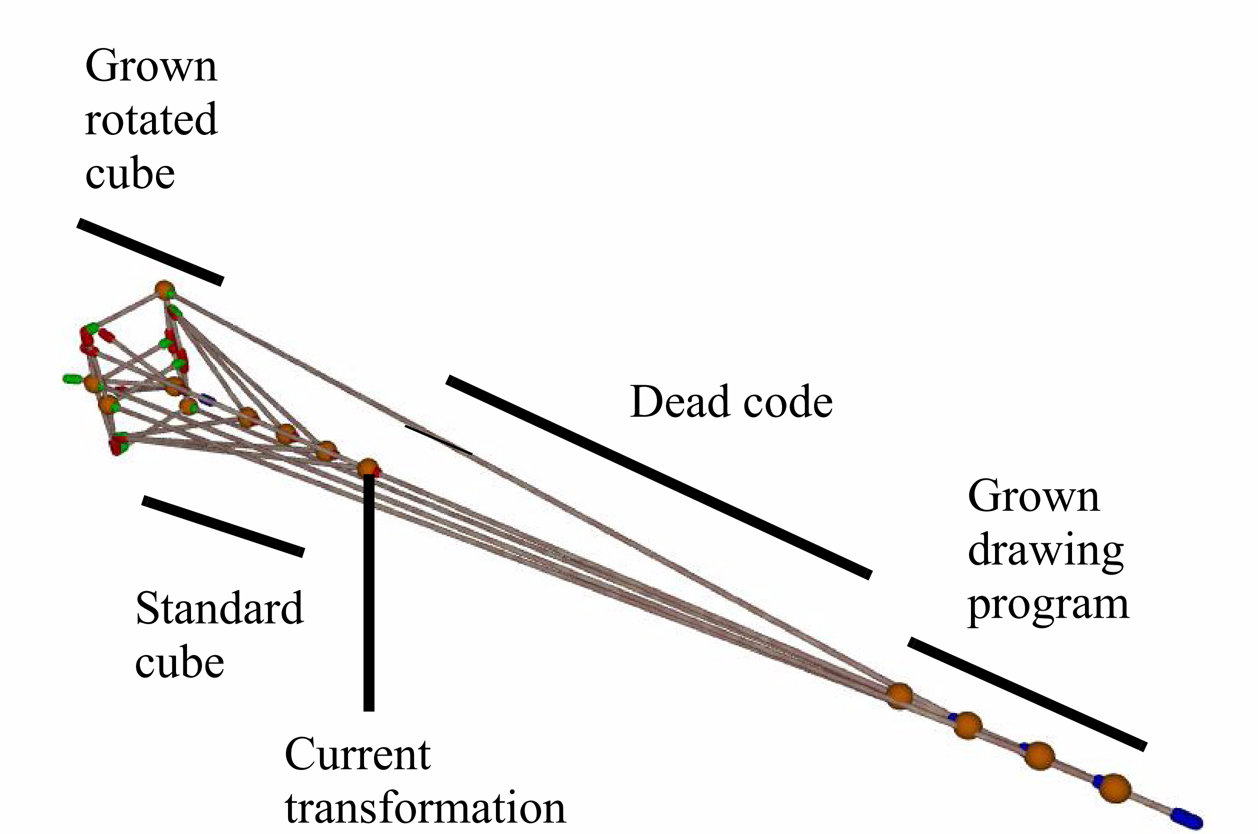
Perspex Programs

- Perspex programs look very much like networks of biological neurons.

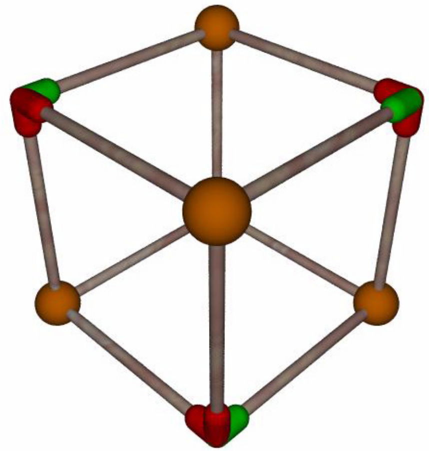


Perspex Programs

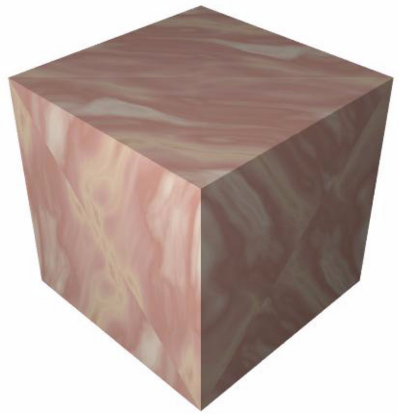
- Perspex programs grow by writing instructions and die by writing H.

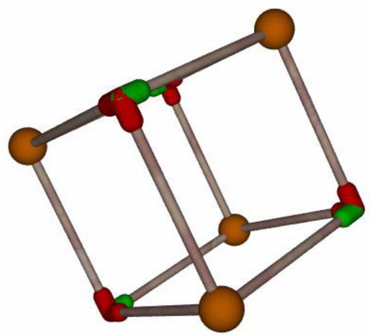


Perspex Rotation Program



Standard cube





Rotated cube

Isolinear Perspex Materials

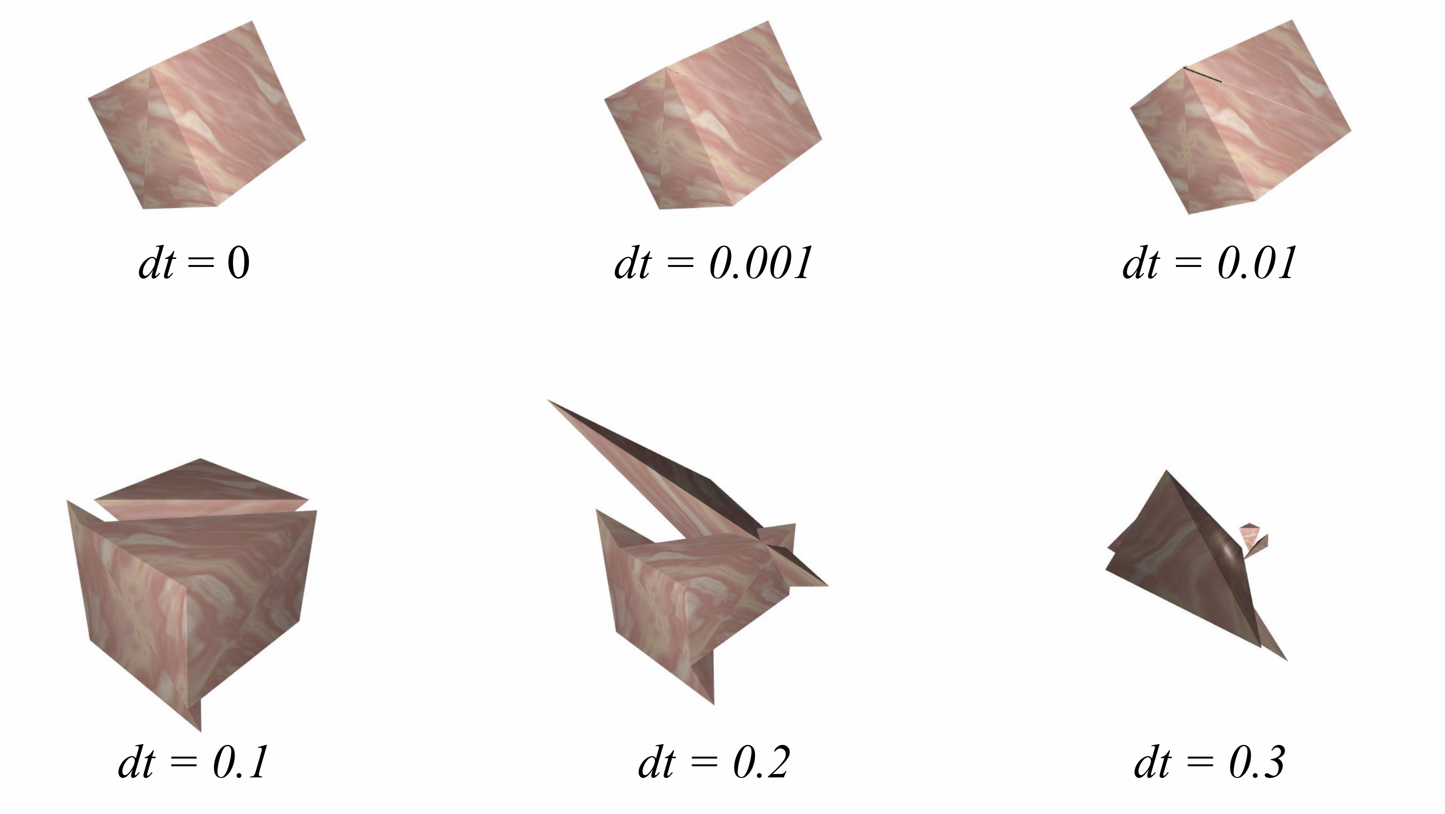
- Turing programs can be laid out as perspexes at integral (integer numbered) locations on a line or on the nodes of an integral lattice in space[5].
- Every point in space can be filled with a *linear* blend of its neighbouring lattice nodes[7]. This produces a continuum of programs filling space.
- Starting the perspex machine at nearby points in this continuum computes in nearly the same way[7]. That is, all of the reads, writes, and jumps are nearly the same.
- The space is an *isomer* of all of the programs that it contains. Hence, the space is called an *isolinear* material.

Isolinear Perspex Materials

- Are robust to errors in the starting point.
- Are robust to missing instructions.
- Can have non-halting Turing programs as singularities in a neighbourhood of programs that are Turing computable and which compute in nearly the same way as the non-halting program – except that they halt!
- Degrade gracefully with increasing errors in starting conditions and make partial recoveries, as predicted by the Walnut Cake Theorem.
- Can be approximated by a sum of band-pass-filter channels. So one instruction can approximate many.

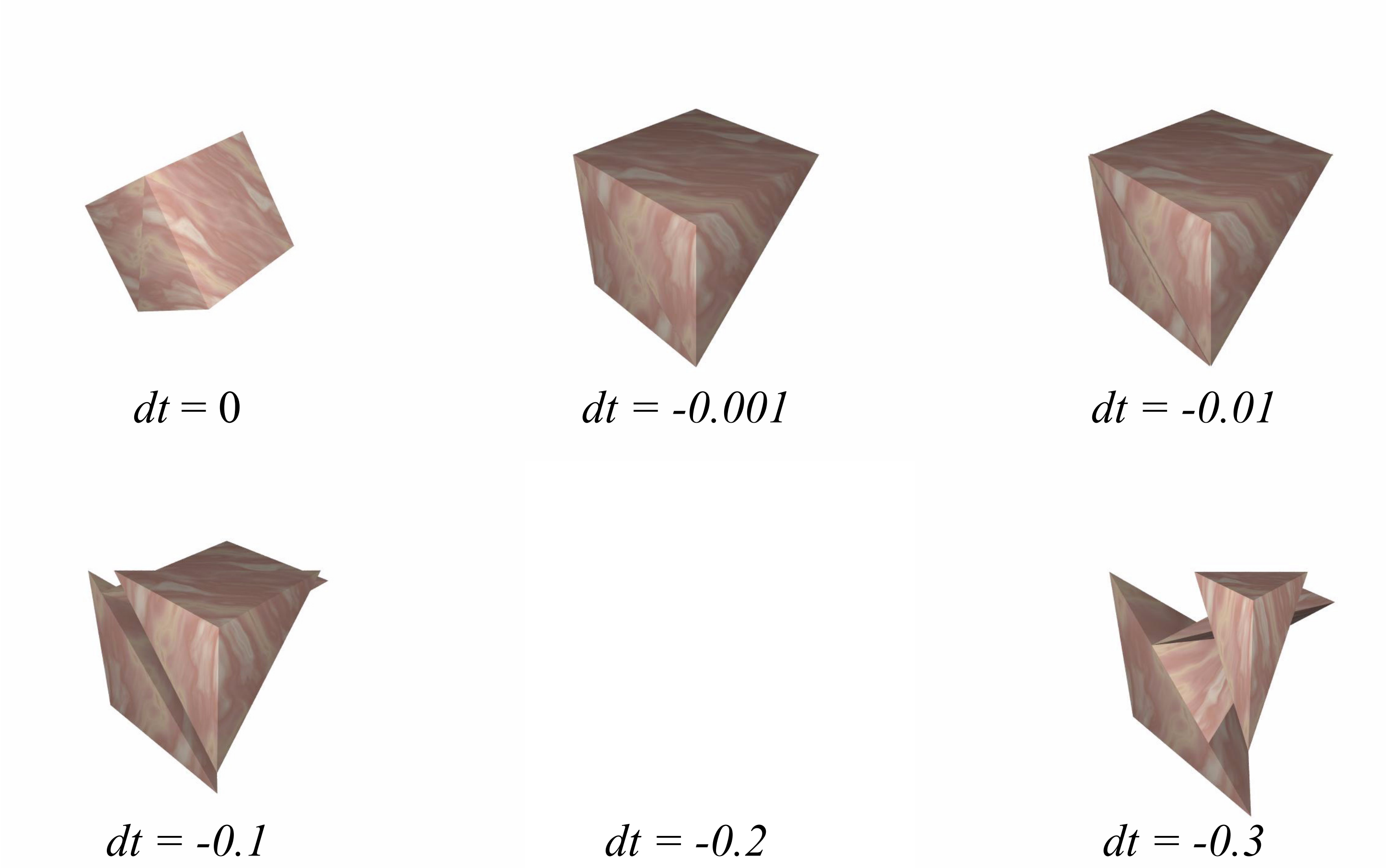
Isolinear Rotation Program

- The program is robust to error, dt, in the starting point.



Isolinear Rotation Program

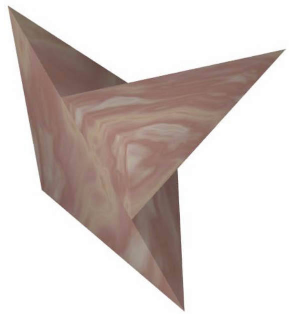
- The program has an isolated non-halting program at $dt = -0.2$ surrounded by computable programs.



Isolinear Rotation Program

- The program partially recovers at $dt = 4 + 0.3$.

dt = 3 - 0.3



dt = 3 +0.3

dt = 4 - 0.3

dt = 4 + 0.3

dt = 5 - 0.3

dt = 5 + 0.3

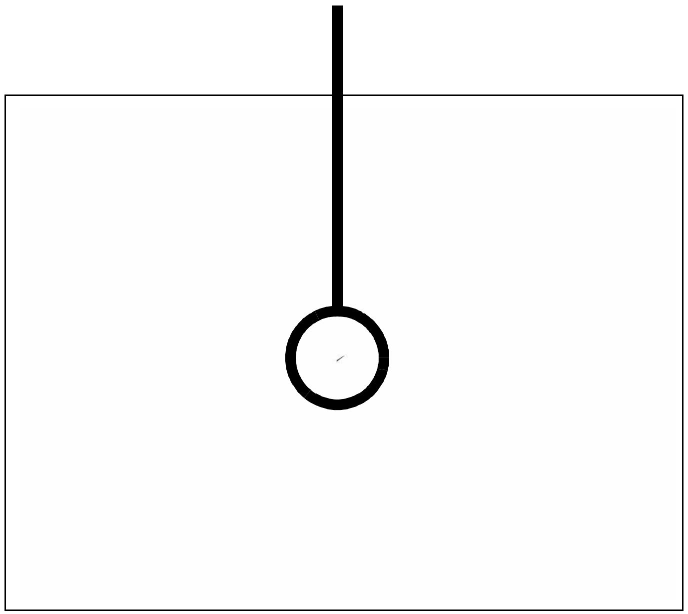
Isostatic Perspex Materials

- If a perspex program is convolved with a triangular filter it gives rise to an isolinear perspex material.
- If a perspex program is convolved with a box filter it gives rise to an *isostatic* perspex material.
- It is easy to analyse the perspex rotation program using box filters and to synthesise the program from the filter channels, or bands.

Isostatic Rotation Program

- Band 4 has 1 perspex in it. It is the DC term. It writes perspexes throughout a time line and, in particular, writes 1 perspex into the $t = 1$ hyperplane where the isolinear rotation program writes its result.

perspex at centre of annulus



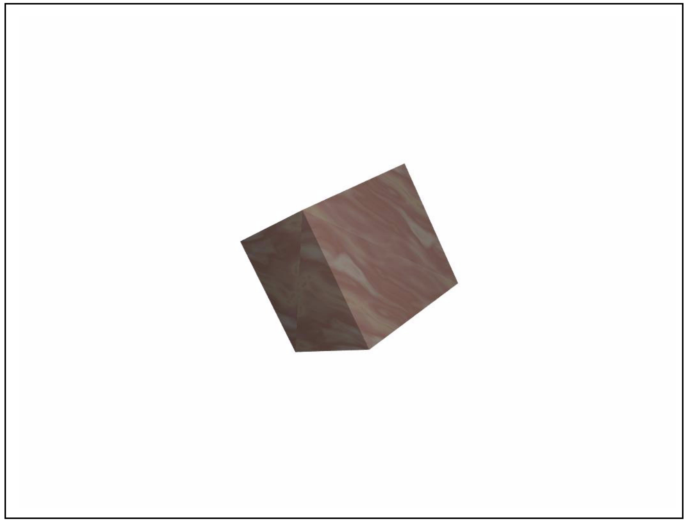
Isostatic Rotation Program

- Band 3 has 2 perspexes. Bands 4 + 3 cause 6 perspexes to be executed, 1 to be written, but 0 are written in the $t = 1$ hyperplane.
- Band 2 has 4 perspexes. Bands 4 + 3 + 2 cause 11 perspexes to be executed, 1 to be written, but 0 are written in the $t = 1$ hyperplane.
- Band 1 has 8 perspexes. Bands 4 + 3 + 2 + 1 are non-halting. That is, they are not Turing computable.

What do you expect to happen when the last band is added?

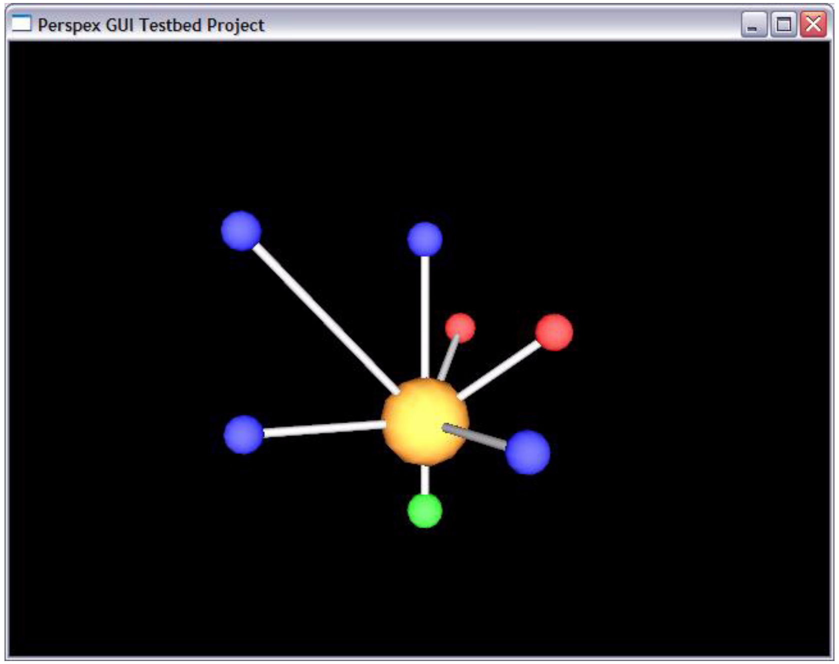
Isostatic Rotation Program

- Band 0 has 16 perspexes. Bands 4 + 3 + 2 + 1 + 0 cause 10 perspexes to be executed, 8 to be written, with 4 written in the $t = 1$ hyperplane.
- The result is identical to the isolinear rotation program with $dt = 0$.



Graphical User Interface

Christopher J.A. Kershaw is developing a simulation of the perspex machine implemented in C^{++} with a graphical user interface implemented in OpenGL.



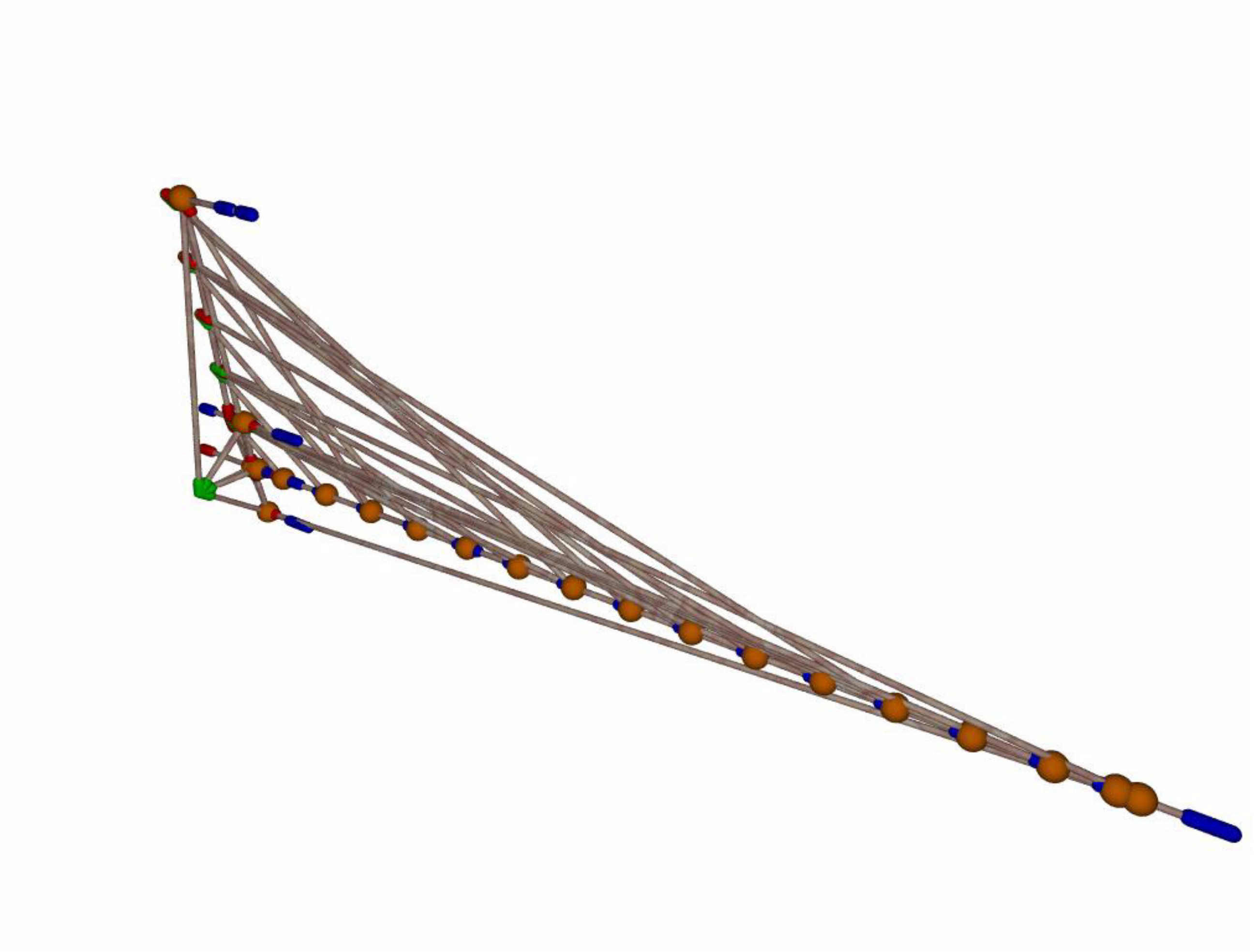
C to Perspex Compiler

Matthew P. Spanner is implementing a C to perspex compiler in Pop11.

C Source

```
a=0;
b=0;
d=5;
for (c=0;c<d;c++){
    a = a + 1;
    b = b + a + c;
}
b=b;
```

Compiled C Source



Philosophy

Randomness

- If a random event is unaffected by earlier or later outcomes then it occurs at a point in time. Reversing time flow so that time passes over that point again gives another opportunity for the event to occur[5].

- If random events and reversals in time flow are a natural part of the universe then they can be detected very simply by measuring the light bouncing off a half-silvered mirror. Some of the light will bounce off the mirror in the “wrong” direction and will move backwards in time with a systematic phase shift[6].

- If this simple experiment is successful, the experimental apparatus is a time machine.

Perspex Thesis

- The perspex machine can model all physical things, including mind, to arbitrary accuracy and, conversely, all physical things, including mind, instantiate a perspex machine.
- The perspex thesis adopts the materialistic thesis that everything that exists is physical.

Good, Evil, Karma, and God

- If the perspex thesis is correct then thoughts are physical things. A good thought can cause a good action in the world and, conversely, a good action in the world can cause a good thought. But goodness of thought and action is just what it means to be good. So good is a physical thing.
- Similarly evil and karma are physical things.
- The idea of God is a physical thing and may cause, and be caused by, actions in the world, but this does not prejudge the question of whether God is a person.

Omniscient God

- A single perspex neuron can encode infinitely many, $\aleph_0$, facts. For example, in a language with n symbols the successive digits of a real number base n can encode all of the produced sentences of the language.

- In this sense, a single perspex neuron can be omniscient. And a mind made of such neurons, if they can exist, can be omniscient. Hence, God may be omniscient.

- If the perspex thesis is correct and God exists then God's mind can be described to arbitrary accuracy by a perspex machine and, conversely, God's mind instantiates a perspex machine.

Mind-Body Problem

- If the perspex thesis is correct then a perspex machine describes physical bodies, and the actions of physical bodies. Conversely, physical bodies, and the actions of physical bodies, instantiate a perspex machine. Thus, the perspex machine is one solution to the mind-body problem.

- The phenomena of consciousness and free will have been defined in perspex formulas[1,3,6]. These formulas can be subjected to philosophical analysis and empirical testing in a robot.

Language and Logic

- Language and logic are symbolic modes of reasoning and are, therefore, less powerful than visualisation of a continuous space.
- Non-verbal animals might be more intelligent than verbal ones.
- It is an open question of physics whether animal brains have access to a continuum and can, thereby, be genuinely non-verbal.

Limits of Language and Logic

- Language and logic apply to a subset of the perspex machine, not to the whole of it. The perspex machine cannot be described entirely in language.
- Logical contradictions, such as non-halting programs, can occur as singularities in a region of space that is everywhere else perspex computable.
- Logically impossible things are generally perspex possible in a region of space asymptotically close to logic.
- If God exists and is omnipotent then God can do anything that a continuous perspex machine can do, if it can exist, so God may do logically impossible things.

Limits of Language and Logic

- Perspex machines that contain noise generally escape from non-halting regions of perspex space. That is, they do not get stuck in non-halting computations. Thus, finite beings derive a practical advantage from fallibility (noise).
- A perspex machine operating in the continuum can solve the halting problem. It can detect Turing incomputabilities and work round them. Thus, infinite beings do not derive any advantage from fallibility.
- If God exists God may be infallible.

Spatial Effects on Mind

The geometry of the perspex machine gives rise to various mental effects:

- Non-monotonicity in learning, reasoning, recovery from illness and injury.
- Paradigm shifts in scientific theories.
- Robustness to missing instructions and graceful degradation, with partial recovery, in the face of increasing deletion of instructions.
- Graceful degradation, with partial recovery, when starting reasoning in the wrong place.

Spatial Effects on Mind

The geometry of the perspex machine also gives rise to:

- Global reasoning.

It is astonishing that these properties arise from the geometry of a perspex machine, or animal brain, and not from the text of a Turing program, or the contents of an animal's thoughts.

Summary

- The perspex machine operates in a continuous space of perspective (or augmented Euclidean) transformations.
- The perspex machine is super-Turing.
- The Turing aspects are stable, but the perspex machine is still being developed to make it easier to access its super-Turing properties.
- Various mental properties are consequences of the geometry of the perspex machine, not of the text or other contents of a program. These are: global reasoning, graceful degradation, paradigm shifts, recovery, relapse, robustness to starting conditions, and robustness to missing instructions.

End Matter

References

1) Anderson, J.A.D.W. “Visual Conviction” *Proceedings of the fifth Alvey Vision Conference* University of Reading, England, 301-303 (1989).

2) Anderson, J.A.D.W. “Representing Geometrical Knowledge” *Phil. Trans. R. Soc. Lond. B.* vol. 352, 1129 - 1140 (1997).

3) J.A.D.W. Anderson “Robot Free Will” *ECAI 2002 Proceedings of the 15th European Conference on Artificial Intelligence* Lyon, France, ed. F. van Harmelan, 559-563 (2002).

References

4) J.A.D.W. Anderson, "Exact Numerical Computation of the Rational General Linear Transformations" in *Vision Geometry XI* Longin Jan Latecki, David M. Mount, Angela Y. Wu, Editors, Proceedings of the SPIE Vol. 4794, 22-28 (2002).

5) J.A.D.W. Anderson, "Perspex Machine" in *Vision Geometry XI* Longin Jan Latecki, David M. Mount, Angela Y. Wu, Editors, Proceedings of the SPIE Vol. 4794, 10-21 (2002).

6) J.A.D.W. Anderson, "Perspex Machine II: Visualisation" in *Vision Geometry XIII* Longin Jan Latecki, David M. Mount, Angela Y. Wu, Editors, Proceedings of the SPIE Vol. 5675, (2005).

References

7) J.A.D.W. Anderson, "Perspex Machine III: Continuity Over the Turing Operations" in *Vision Geometry XIII* Longin Jan Latecki, David M. Mount, Angela Y. Wu, Editors, Proceedings of the SPIE Vol. 5675, (2005).

Acknowledgements

I would like to thank the following undergraduates:

- Christopher J.A. Kershaw for a Graphical User Interface to a perspex machine implemented in C^{++}.
- Matthew P. Spanner for examples from a C to perspex compiler.