The Future of Personal Computing System Construction

Yoshiki Ohshima

Viewpoints Research Institute
SAP CDG

AGERE 2013

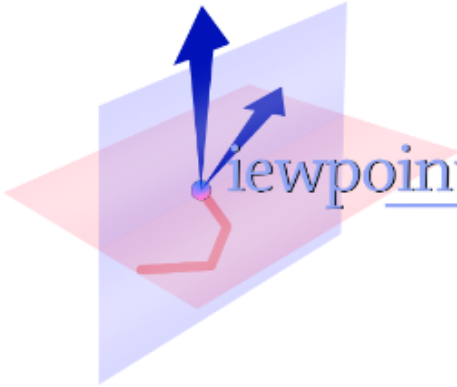
Indianapolis, October 27th, 2013

Who we are

Who we are


Alan Kay


Kim Rose



Viewpoints Research Institute

Who we are


Alan Kay


Kim Rose

Viewpoints Research Institute

Computing in Education

Who we are


Alan Kay


Kim Rose

Computing in Education


Squeak Etoys

Who we are


Alan Kay


Kim Rose

Viewpoints Research Institute

Computing in Education

Who we are


Alan Kay


Kim Rose

Viewpoints Research Institute

Computing in Education

Who we are


Alan Kay


Kim Rose

Viewpoints Research Institute

Computing in Education

Research on Computing

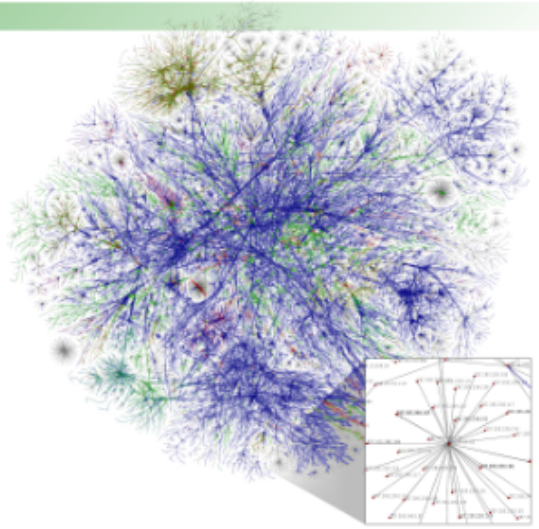
The STEPS project
CDG at SAP

SAP®

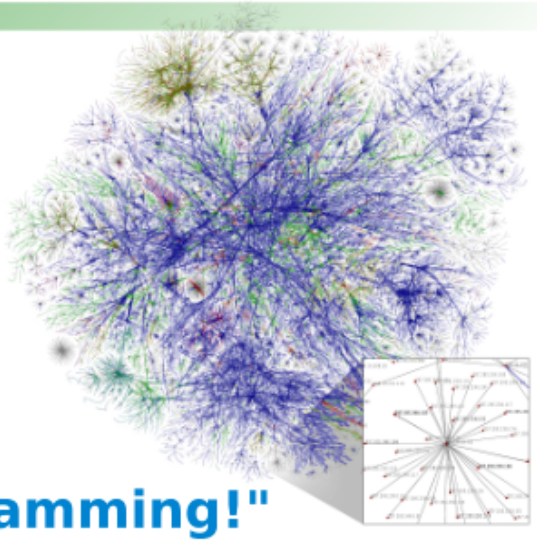
The STEPS Project

The STEPS Project

A large NSF funded project:
"Reinventing the Internet"



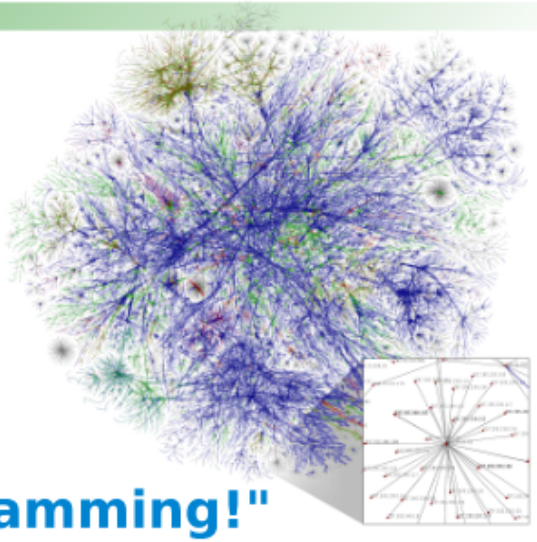
The STEPS Project



A large NSF funded project:
"Reinventing the Internet"

"We need to reinvent programming!"

The STEPS Project

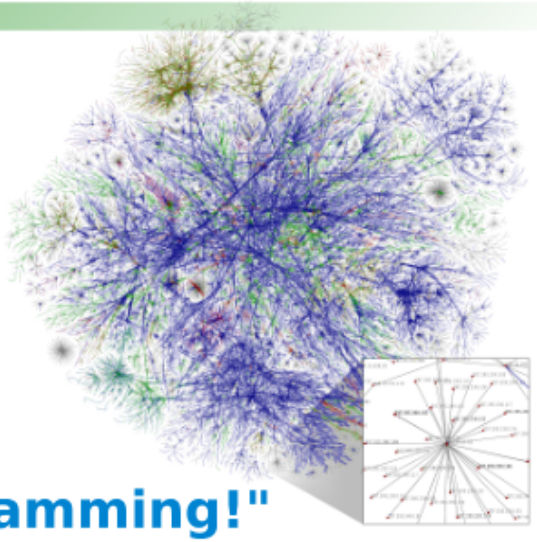


A large NSF funded project:
"Reinventing the Internet"

"We need to reinvent programming!"

NSF: "please turn it in as a proposal"

The STEPS Project



A large NSF funded project:
"Reinventing the Internet"

"We need to reinvent programming!"

NSF: "please turn it in as a proposal"

The STEPS Project (2007-2013)

Code Seems “Large” and “Complicated” For What It Does

a 400 page book
- 20,000 lines

1 foot of books
(without covers)
15 books
6000 pages
300,000 lines

1 million LOC/meter!

Code Seems “Large” and “Complicated” For What It Does

a 400 page book
- 20,000 lines

1 foot of books
(without covers)
15 books
6000 pages
300,000 lines

1 million LOC/meter!

Empire State Building
- **1472 feet**
- **22080 books**
- **8,832,000 pages**
- **441,600,000 lines**

Code Seems “Large” and “Complicated” For What It Does

a 400 page book
- 20,000 lines

1 foot of books
(without covers)
15 books
6000 pages
300,000 lines

1 million LOC/meter!

Empire State Building
- 1472 feet
- 22080 books
- 8,832,000 pages
- 441,600,000 lines

“Personal Computing” ?

- ~220,000,000 lines of code

- 11,000 books

Office Apps:
- ~100,000,000 lines of code
- 5,000 books

Operating System:
- ~120,000,000+ lines of code
- 6,000 books

Code Seems “Large” and “Complicated” For What It Does

a 400 page book
- 20,000 lines

1 foot of books
(without covers)
15 books
6000 pages
300,000 lines

1 million LOC/meter!

Empire State Building
- 1472 feet
- 22080 books
- 8,832,000 pages
- 441,600,000 lines

“Personal Computing” ?

- ~220,000,000 lines of code

- 11,000 books

Office Apps:
- ~100,000,000 lines of code
- 5,000 books

LFP Company:
~400,000,000 lines of code
~170,000 screens
~20,000 books

Operating System:
- ~120,000,000+ lines of code
- 6,000 books

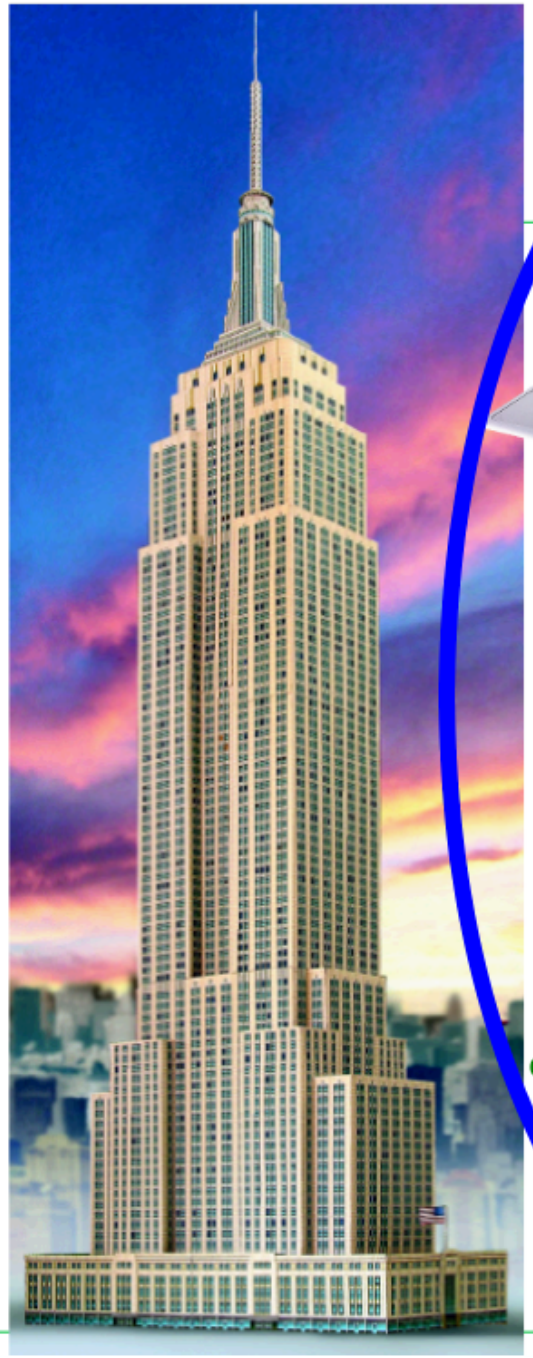
Code Seems "Large" and "Complicated" For What It Does

a 400 page book
- 20,000 lines

1 foot of books
(without covers)
15 books
6000 pages
300,000 lines

1 million LOC/meter!

Empire State Building
- 1472 feet
- 22080 books
- 8,832,000 pages
- 441,600,000 lines



"Personal Computing" ?

- ~220,000,000 lines of code

- 11,000 books

Office Apps:
- ~100,000,000 lines of code
- 5,000 books

LFP Company:
~400,000,000 lines of code
~170,000 screens
~20,000 books

Operating System:
- ~120,000,000+ lines of code
- 6,000 books

The STEPS Project

Make a personal computing environment with a concise and understandable representation.

~ 20,000 LOC, maybe?

The STEPS Project

Make a personal computing environment
with a concise and understandable representation.
~ 20,000 LOC, maybe?

Squeak Smalltalk System: 400,000 LOC
- experiments, dead code, optimization (2x)
- we would just need 10x reduction?

The STEPS Project

Make a personal computing environment with a concise and understandable representation.

~ 20,000 LOC, maybe?

Squeak Smalltalk System: 400,000 LOC

- experiments, dead code, optimization (2x)

- we would just need 10x reduction?

Design DSLs to describe different parts of the system clearly.

The Frankenstein's Monster Model!

The Frankenstein's Monster Model!

The Frankenstein's Monster Model!

Execution Engine

The Frankenstein's Monster Model!

Graphics System

Execution Engine

The Frankenstein's Monster Model!

App Framework

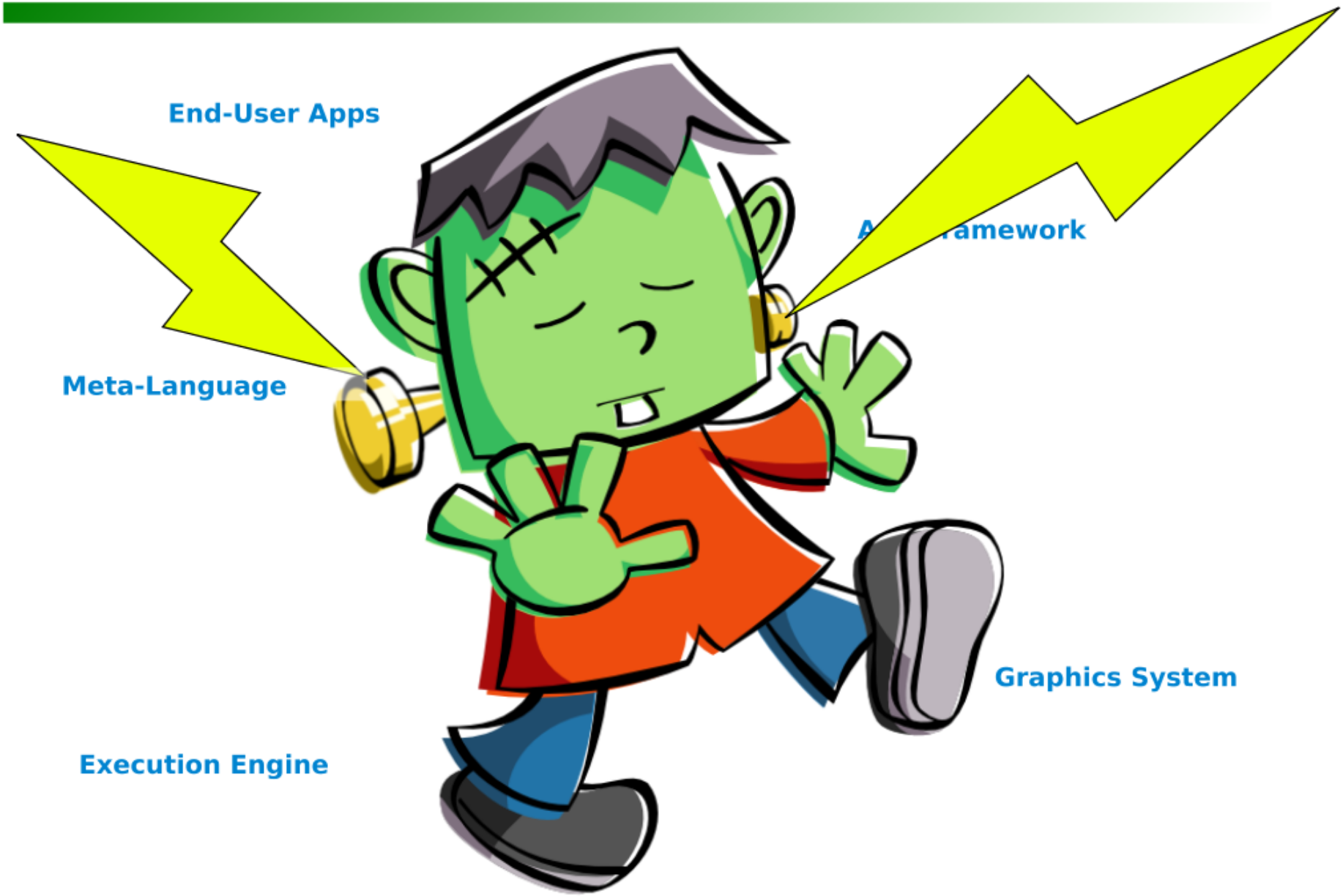
Graphics System

Execution Engine

The Frankenstein's Monster Model!

The Frankenstein's Monster Model!

The Frankenstein's Monster Model!



The Frankenstein's Monster Model!

The Frankenstein's Monster Model!

End-User Apps

App Framework

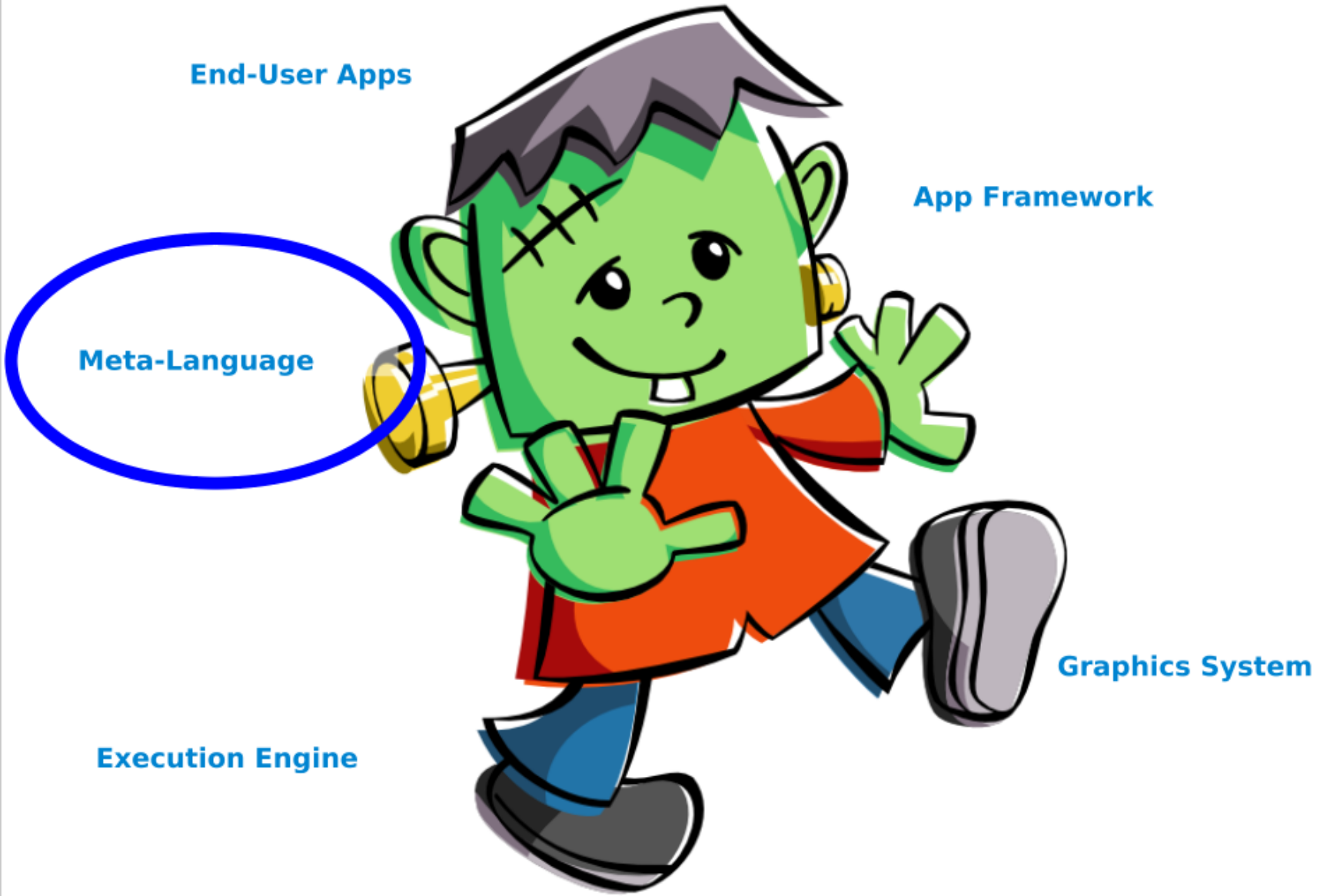
Meta-Language

Graphics System

Execution Engine

(We wrote it all in under 20,000 lines)

A Meta-Language



Alex Warth

OMeta Can Make Symbol Transformers

1 + 2 / 3 - 5

Do It

```
OMeta2 subclass: OMeta2Calculator ()

start = addExpr:e spaces                    [e].

addExpr =
      addExpr:x ("+" | "-"):op mulExpr:y    [{op. x. y}]
|     mulExpr.

mulExpr =
      mulExpr:x ("*" | "/"):op primExpr:y   [{op. x. y}]
|     primExpr.

primExpr =
      "(" addExpr:x ")"                     [x]
|     number
|     identifier.

number =
      spaces <digit+>.

identifier =
      spaces <letter (letter | digit)*>.
```

(a / b + 35) * (6 + m) / a * P

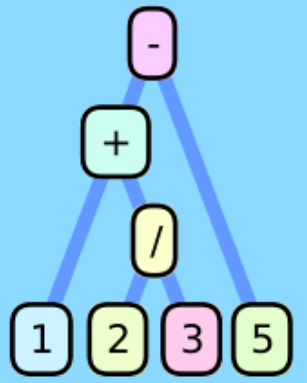
Alex Warth

OMeta Can Make Symbol Transformers

1 + 2 / 3 - 5

Do It

```
OMeta2 subclass: OMeta2Calculator ()

start = addExpr:e spaces                        [e].

addExpr =
      addExpr:x ("+" | "-"):op mulExpr:y        [{op. x. y}]
|     mulExpr.

mulExpr =
      mulExpr:x ("*" | "/"):op primExpr:y       [{op. x. y}]
|     primExpr.

primExpr =
      "(" addExpr:x ")"                         [x]
|     number
|     identifier.

number =
      spaces <digit+>.

identifier =
      spaces <letter (letter | digit)*>.
```



(a / b + 35) * (6 + m) / a * P

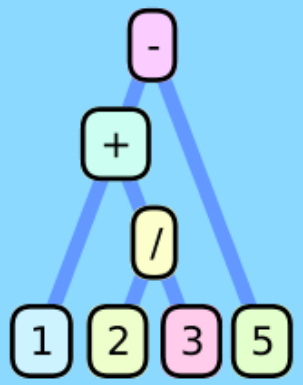
Alex Warth

OMeta Can Make Symbol Transformers

(a / b + 35) *
(6 + m) / a * P

Do It

```
OMeta2 subclass: OMeta2Calculator ()

start = addExpr:e spaces                    [e].

addExpr =
      addExpr:x ("+" | "-"):op mulExpr:y    [{op. x. y}]
|     mulExpr.

mulExpr =
      mulExpr:x ("*" | "/"):op primExpr:y   [{op. x. y}]
|     primExpr.

primExpr =
      "(" addExpr:x ")"                     [x]
|     number
|     identifier.

number =
      spaces <digit+>.

identifier =
      spaces <letter (letter | digit)*>.
```



(a / b + 35) * (6 + m) / a * P

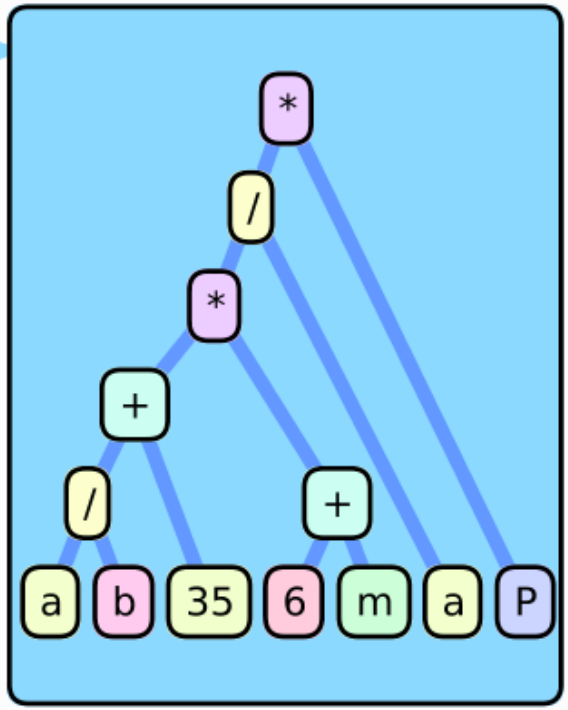
Alex Warth

OMeta Can Make Symbol Transformers

```
(a / b + 35) *
(6 + m) / a * P
```

Do It

```
OMeta2 subclass: OMeta2Calculator ()

start = addExpr:e spaces                          [e].

addExpr =
      addExpr:x ("+" | "-"):op mulExpr:y          [{op. x. y}]
|     mulExpr.

mulExpr =
      mulExpr:x ("*" | "/"):op primExpr:y         [{op. x. y}]
|     primExpr.

primExpr =
      "(" addExpr:x ")"                           [x]
|     number
|     identifier.

number =
      spaces <digit+>.

identifier =
      spaces <letter (letter | digit)*>.
```



(a / b + 35) * (6 + m) / a * P

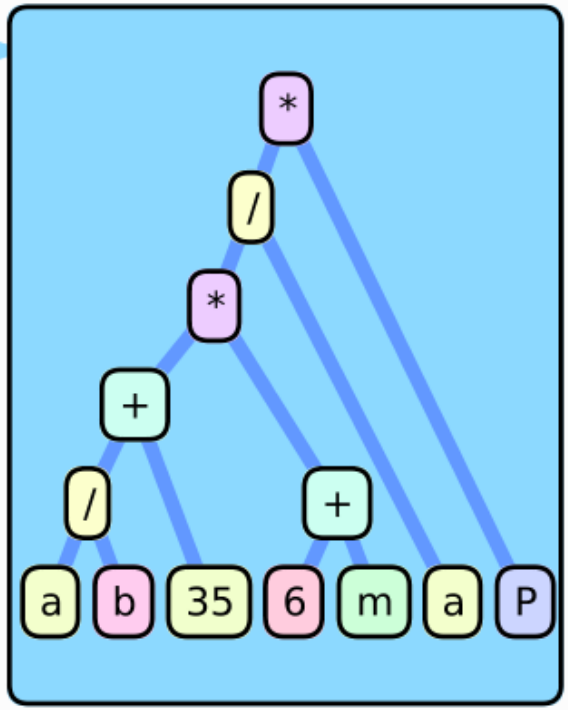
Alex Warth

OMeta Can Make Symbol Transformers

(a / b + 35.5) * (6 + m) / a * P

Do It

```
OMeta2 subclass: OMeta2Calculator ()

start = addExpr:e spaces                          [e].

addExpr =
      addExpr:x ("+" | "-"):op mulExpr:y          [{op. x. y}]
|     mulExpr.

mulExpr =
      mulExpr:x ("*" | "/"):op primExpr:y         [{op. x. y}]
|     primExpr.

primExpr =
      "(" addExpr:x ")"                           [x]
|     number
|     identifier.

number =
      spaces <digit+>.

identifier =
      spaces <letter (letter | digit)*>.
```



(a / b + 35) * (6 + m) / a * P

Alex Warth

OMeta Can Make Symbol Transformers

(a / b + 35.5) * (6 + m) / a * P

Do It

```
OMeta2 subclass: OMeta2Calculator ()

start = addExpr:e spaces                    [e].

addExpr =
      addExpr:x ("+" | "-"):op mulExpr:y    [{op. x. y}]
|     mulExpr.

mulExpr =
      mulExpr:x ("*" | "/"):op primExpr:y   [{op. x. y}]
|     primExpr.

primExpr =
      "(" addExpr:x ")"                     [x]
|     number
|     identifier.

number =
      spaces <digit+>.

identifier =
      spaces <letter (letter | digit)*>.
```

error occured

OK

(a / b + 35) * (6 + m) / a * P

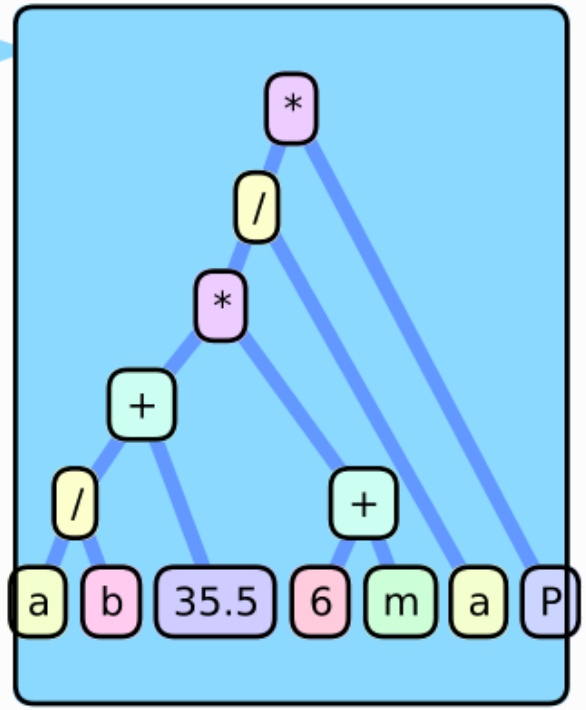
Alex Warth

OMeta Can Make Symbol Transformers

(a / b + 35.5) * (6 + m) / a * P

Do It

```
OMeta2 subclass: OMeta2Calculator ()
|
start = addExpr:e spaces                    [e].

addExpr =
      addExpr:x ("+" | "-"):op mulExpr:y    [{op. x. y}]
|     mulExpr.

mulExpr =
      mulExpr:x ("*" | "/"):op primExpr:y   [{op. x. y}]
|     primExpr.

primExpr =
      "(" addExpr:x ")"                     [x]
|     number
|     identifier.

number =
      spaces <digit+ ("." digit+)? >.

identifier =
      spaces <letter (letter | digit)*>.
```



(a / b + 35) * (6 + m) / a * P

Alex Warth

OMeta Can Manipulate Trees

```
{'-'.
 {'+'.
  '1'.
  {'/'. '4'. '2'}}.
 '5'}
```

```
{'a' -> 5.
 'b' -> 10}
```

Do It

```
OMeta2 subclass: OMeta2Evaluator (env)

start :env = calc.

calc =
    {('+' | '-' | '*' | '/'):op calc:l calc:r}
                    -> [l perform: op asSymbol with: r]
|   _:n ?[n first isDigit]
                    -> [n asNumber]
|   _:n ?[n first isLetter]
                    -> [env at: n ifAbsent: [Float nan]].
```

Alex Warth

OMeta Can Manipulate Trees

```
{'-'.
 {'+'.
  '1'.
  {'/'. '4'. '2'}}.
 '5'}
```

```
{'a' -> 5.
 'b' -> 10}
```

Do It

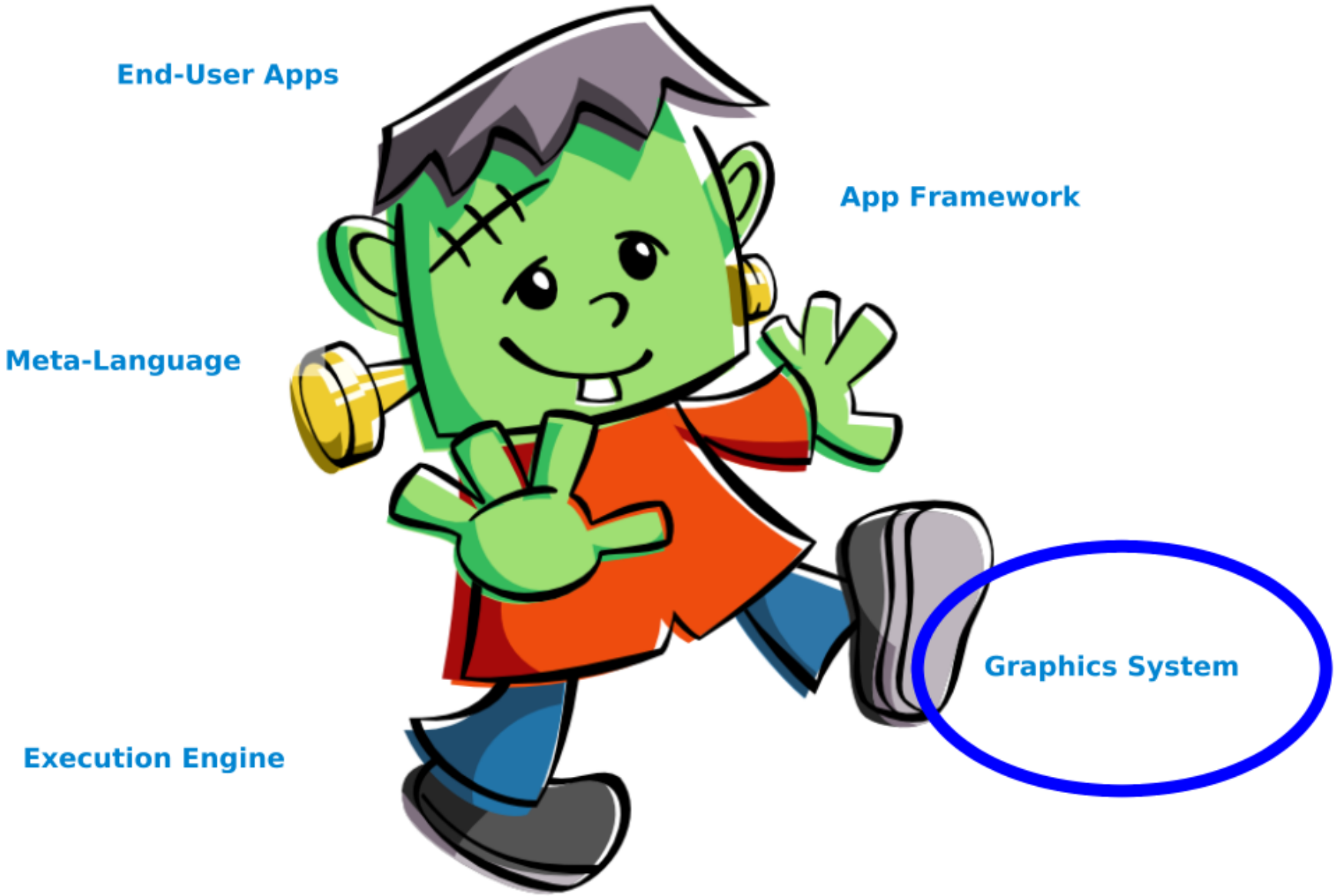
```
OMeta2 subclass: OMeta2Evaluator (env)

start :env = calc.

calc =
    {('+' | '-' | '*' | '/'):op calc:l calc:r}
                    -> [l perform: op asSymbol with: r]
|   _:n ?[n first isDigit]
                    -> [n asNumber]
|   _:n ?[n first isLetter]
                    -> [env at: n ifAbsent: [Float nan]].
```

-2

It really is a Meta-Language!

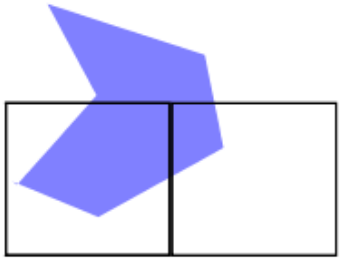
```
Ometa2 subclass: Ometa2RuleParser ()
 nameFirst = letter.
 nameRest  = nameFirst | digit.

 nsName =
    firstAndRest(#nameFirst. #nameRest):xs -> [(String withAll: xs) asSymbol]
 |  $_ -> [#anything].

 optIter :x =
    "*"       -> [{#Many.  x}]
|   "+"       -> [{#Many1. x}]
|   "?" ~$[   -> [{#Opt. x}]
|   empty         -> [x].

expr1 =
    (keyword('true') | keyword('false') | keyword('nil')):lit -> [{#App. #exactly. lit}]
|   application | semanticAction | semanticPredicate | characters | tokenSugar
|   stringLiteral |   symbolLiteral |   numberLiteral |   characterLiteral
|   "{"  expr:e "}" -> [{#Form. e}]
|   "<"  expr:e ">" -> [{#ConsBy. e}]
|   "("  expr:e ")" -> [e].

rule =
    &(^space* nsName):n rulePart(n):x ("," rulePart(n))*:xs spaces end
         -> [{#Rule. n. {#Or. x}, xs}].

rulePart :ruleName =
    name:n ?[n = ruleName] expr4:b1
         (    "=" expr:b2   -> [{#And. b1. b2}]
         |    empty              -> [b1]).

grammar =
    name:s "subclass:" name:n "(" < name* >:vs ")"
    (< rule >:r "." [r])+:rs spaces.
```

A Graphics System



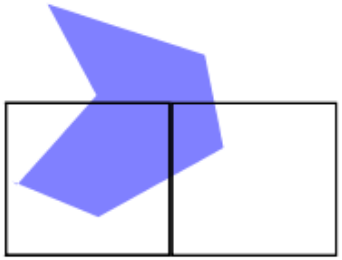
Dan Amelang

Nile Dataflow Language and Gezira Graphics Engine



Nile Dataflow Language and Gezira Graphics Engine

Dan Amelang



$$\sigma(P,Q) = (Q_y - P_y)(x + 1 - \frac{Q_x + P_x}{2})$$

$$\begin{aligned} \gamma(P) &= min(x+1, max(x, P_x)), \\ & \quad min(y+1, max(y, P_y)) \end{aligned}$$

$$\begin{aligned} \omega(P) &= \frac{1}{m}(\gamma(P)_y - P_y) + P_x, \\ & \quad m(\gamma(P)_x - P_x) + P_y \end{aligned}$$

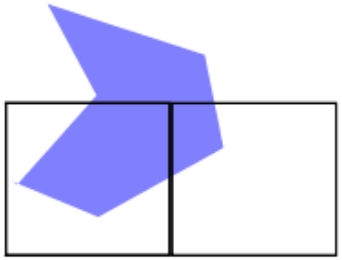
$$\begin{aligned} coverage(\overrightarrow{AB}) &= \sigma(\gamma(A), \gamma(\omega(A))) + \\ & \quad \sigma(\gamma(\omega(A)), \gamma(\omega(B))) + \\ & \quad \sigma(\gamma(\omega(B)), \gamma(B)) \end{aligned}$$

$$min(|\sum coverage(\overrightarrow{AB}_i)|, 1)$$

"The Formula"

Nile Dataflow Language and Gezira Graphics Engine

Dan Amelang



Antialiased Vector Rendering

~ 45 LOC

$$\sigma(P,Q) = (Q_y - P_y)(x + 1 - \frac{Q_x + P_x}{2})$$

$$\gamma(P) = min(x+1, max(x, P_x)), \; min(y+1, max(y, P_y))$$

$$\omega(P) = \frac{1}{m}(\gamma(P)_y - P_y) + P_x, \; m(\gamma(P)_x - P_x) + P_y$$

$$coverage(\overrightarrow{AB}) = \sigma(\gamma(A), \gamma(\omega(A))) + \sigma(\gamma(\omega(A)), \gamma(\omega(B))) + \sigma(\gamma(\omega(B)), \gamma(B))$$

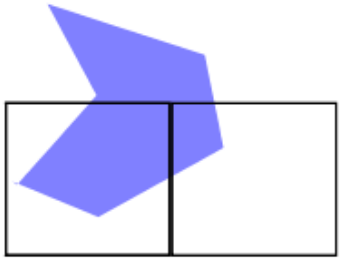
$$min(|\sum coverage(\overrightarrow{AB}_i)|, 1)$$

"The Formula"

```
type Point = (x, y : Real)
type Bezier = (A, B, C : Point)
type EdgeSpan = (x, y, c, l : Real)
type EdgeSample = (x, y, a, h : Real)

| (a : Real) | : Real
   { -a if a < 0, a }

(a : Real) ◁ (b : Real) : Real
   { a if a < b, b }

(a : Real) ~ (b : Real) : Real
   (a + b) / 2

DecomposeBeziers : Bezier >> EdgeSample
   ∀ (A, B, C)
       inside = (⌊ A ⌋ = ⌊ C ⌋ ∨ ⌈ A ⌉ = ⌈ C ⌉)
       if inside.x ∧ inside.y
           P = ⌊ A ⌋ ◁ ⌊ C ⌋
           w = P.x + 1 - (C.x ~ A.x)
           h = C.y - A.y
           >> (P.x + 1/2, P.y + 1/2, w × h, h)
       else
           ABBC    = (A ~ B) ~ (B ~ C)
           min     = ⌊ ABBC ⌋
           max     = ⌈ ABBC ⌉
           nearmin = | ABBC - min | < 0.1
           nearmax = | ABBC - max | < 0.1
           M       = {min if nearmin, max if nearmax, ABBC}
           << (M, B ~ C, C) << (A, A ~ B, M)

CombineEdgeSamples : EdgeSample >> EdgeSpan
   (x, y, A, H) = 0
   ∀ (x', y', a, h)
       if y' = y
           if x' = x
               A' = A + a
               H' = H + h
           else
               l = {x' - x - 1 if |H| > 0.5, 0}
               >> (x, y, |A| ◁ 1, l)
               A' = H + a
               H' = H + h
       else
           >> (x, y, |A| ◁ 1, 0)
           A' = a
           H' = h
   >> (x, y, |A| ◁ 1, 0)

Rasterize : Bezier >> EdgeSpan
   ⇒ DecomposeBeziers ⇒ SortBy (@x)
     ⇒ SortBy (@y) ⇒ CombineEdgeSamples
```

Nile Dataflow Language and Gezira Graphics Engine

Dan Amelang



Antialiased Vector Rendering

~ 45 LOC

This seems like a lot (the current commercial examples and operating systems require 10s of millions to 100s of millions of lines of code).

We ask: "How complex is this **really**?" Could *active mathematics* be invented to represent the semantic essence of "personal computing from the end-user down to the metal?"

Could we produce runnable code that is many orders of magnitude smaller? A few "100s of 'Maxwell's Equations' T-shirts" vs. 10s of millions of pages of code?

$$\sigma(P,Q) = (Q_y - P_y)(x + 1 - \frac{Q_x + P_x}{2})$$

$$\gamma(P) = min(x+1, max(x, P_x)), \; min(y+1, max(y, P_y))$$

$$\omega(P) = \frac{1}{m}(\gamma(P)_y - P_y) + P_x, \; m(\gamma(P)_x - P_x) + P_y$$

$$coverage(\overrightarrow{AB}) = \sigma(\gamma(A), \gamma(\omega(A))) + \sigma(\gamma(\omega(A)), \gamma(\omega(B))) + \sigma(\gamma(\omega(B)), \gamma(B))$$

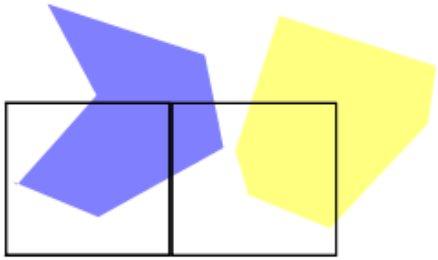
$$min(|\sum coverage(\overrightarrow{AB}_i)|, 1)$$

"The Formula"

```
type Point = (x, y : Real)
type Bezier = (A, B, C : Point)
type EdgeSpan = (x, y, c, l : Real)
type EdgeSample = (x, y, a, h : Real)

| (a : Real) | : Real
   { -a if a < 0, a }

(a : Real) ◁ (b : Real) : Real
   { a if a < b, b }

(a : Real) ~ (b : Real) : Real
   (a + b) / 2

DecomposeBeziers : Bezier >> EdgeSample
   ∀ (A, B, C)
       inside = (⌊ A ⌋ = ⌊ C ⌋ ∨ ⌈ A ⌉ = ⌈ C ⌉)
       if inside.x ∧ inside.y
           P = ⌊ A ⌋ ◁ ⌊ C ⌋
           w = P.x + 1 - (C.x ~ A.x)
           h = C.y - A.y
           >> (P.x + 1/2, P.y + 1/2, w × h, h)
       else
           ABBC    = (A ~ B) ~ (B ~ C)
           min     = ⌊ ABBC ⌋
           max     = ⌈ ABBC ⌉
           nearmin = | ABBC - min | < 0.1
           nearmax = | ABBC - max | < 0.1
           M       = {min if nearmin, max if nearmax, ABBC}
           << (M, B ~ C, C) << (A, A ~ B, M)

CombineEdgeSamples : EdgeSample >> EdgeSpan
   (x, y, A, H) = 0
   ∀ (x', y', a, h)
       if y' = y
           if x' = x
               A' = A + a
               H' = H + h
           else
               l = {x' - x - 1 if |H| > 0.5, 0}
               >> (x, y, |A| ◁ 1, l)
               A' = H + a
               H' = H + h
       else
           >> (x, y, |A| ◁ 1, 0)
           A' = a
           H' = h
   >> (x, y, |A| ◁ 1, 0)

Rasterize : Bezier >> EdgeSpan
   ⇒ DecomposeBeziers ⇒ SortBy (@x)
     ⇒ SortBy (@y) ⇒ CombineEdgeSamples
```

Nile Dataflow Language and Gezira Graphics Engine

Dan Amelang



Antialiased Vector Rendering

~ 45 LOC

Compositing

The 26 compositing ops used by Flash/SVG, Core Graphics, WPF

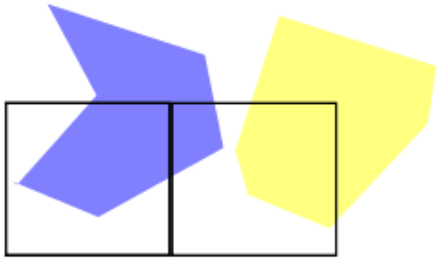
95 LOC

```
(a : Color) ⊕ (b : Color) : Color
    a × (1 - b.a) + b × (1 - a.a)

CompositeClear : Compositor
    ∀ (a, b)
        >> 0

CompositeSrc : Compositor
    ∀ (a, b)
        >> a

CompositeDst : Compositor
    ∀ (a, b)
        >> b

CompositeOver : Compositor
    ∀ (a, b)
        >> a + b × (1 - a.a)

CompositeDstOver : Compositor
    ∀ (a, b)
        >> b + a × (1 - b.a)

CompositeSrcIn : Compositor
    ∀ (a, b)
        >> a × b.a

CompositeDstIn : Compositor
    ∀ (a, b)
        >> b × a.a

CompositeSrcOut : Compositor
    ∀ (a, b)
        >> a × (1 - b.a)

CompositeDstOut : Compositor
    ∀ (a, b)
        >> b × (1 - a.a)

CompositeSrcAtop : Compositor
    ∀ (a, b)
        >> a × b.a + b × (1 - a.a)

CompositeDstAtop : Compositor
    ∀ (a, b)
        >> b × a.a + a × (1 - b.a)

CompositeXor : Compositor
    ∀ (a, b)
        >> a ⊕ b

CompositePlus : Compositor
    ∀ (a, b)
        >> (a + b) ◁ 1
```

```
CompositeOverlay : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
        >> {c if 2 × b < b.a, d}

CompositeDarken : Compositor
    ∀ (a, b)
        >> (a × b.a) ◁ (b × a.a) + (a ⊕ b)

CompositeLighten : Compositor
    ∀ (a, b)
        >> (a × b.a) ▷ (b × a.a) + (a ⊕ b)

CompositeColorDodge : Compositor
    ∀ (a, b)
        c = a.a × b.a + (a ⊕ b)
        d = (b × a.a / (1 - a / a.a) + (a ⊕ b)) ◁ 1
        >> {c if a × b.a + b × a.a ≥ a.a × b.a, d}

CompositeColorBurn : Compositor
    ∀ (a, b)
        c = a.a × (a × b.a + b × a.a - a.a × b.a) / a + (a ⊕ b)
        >> {a ⊕ b if a × b.a + b × a.a ≤ a.a × b.a, c}

CompositeHardLight : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
    >> {c if 2 × a < a.a, d}

CompositeSoftLight : Compositor
    ∀ (a, b)
        c = (1 - b / b.a) × (2 × a - a.a)
        d = b × (a.a - c) + (a ⊕ b)
        e = b × (a.a - c × (3 - 8 × b / b.a)) + (a ⊕ b)
        f = b × a.a + (√(b / b.a) × b.a - b) × (2 × a - a.a) + (a ⊕ b)
        >> {d if 2 × a < a.a, e if b × 8 ≤ b.a, f}

CompositeDifference : Compositor
    ∀ (a, b)
        c = a + b - 2 × ((a × b.a) ◁ (b × a.a))
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeExclusion : Compositor
    ∀ (a, b)
        c = a × b.a + b × a.a - 2 × a × b + (a ⊕ b)
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeSubtract : Compositor
    ∀ (a, b)
        >> (a + b - 1) ▷ 0

CompositeInvert : Compositor
```

Nile Dataflow Language and Gezira Graphics Engine

Dan Amelang



Antialiased Vector Rendering

~ 45 LOC

Compositing

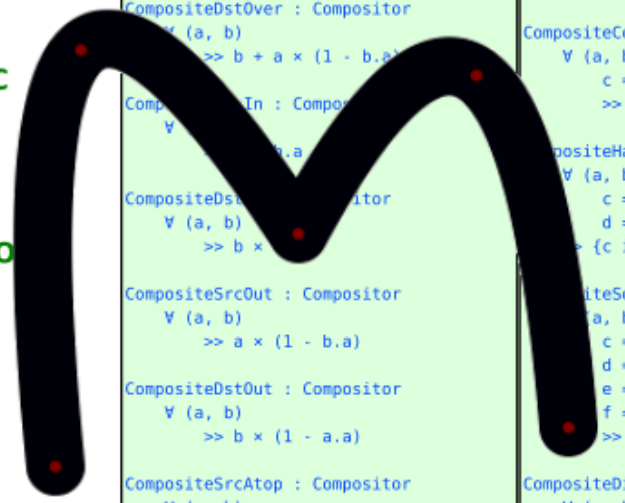
The 26 compositing ops
used by Flash/SVG,
Core Graphics, WPF

95 LOC

Pen Stroking

For outlines and drawings
3 kinds of caps and joins, et

95 LO

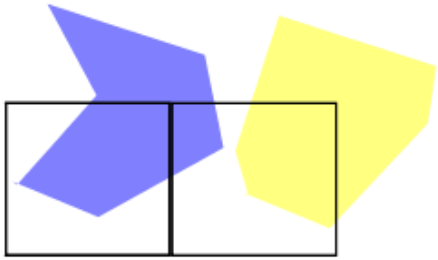


```
(a : Color) ⊕ (b : Color) : Color
    a × (1 - b.a) + b × (1 - a.a)

CompositeClear : Compositor
    ∀ (a, b)
        >> 0

CompositeSrc : Compositor
    ∀ (a, b)
        >> a

CompositeDst : Compositor
    ∀ (a, b)
        >> b

CompositeOver : Compositor
    ∀ (a, b)
        >> a + b × (1 - a.a)

CompositeDstOver : Compositor
    ∀ (a, b)
        >> b + a × (1 - b.a)

Comp        In : Compo
    ∀
          .a

CompositeDs        itor
    ∀ (a, b)
        >> b ×

CompositeSrcOut : Compositor
    ∀ (a, b)
        >> a × (1 - b.a)

CompositeDstOut : Compositor
    ∀ (a, b)
        >> b × (1 - a.a)

CompositeSrcAtop : Compositor
    ∀ (a, b)
        >> a × b.a + b × (1 - a.a)

CompositeDstAtop : Compositor
    ∀ (a, b)
        >> b × a.a + a × (1 - b.a)

CompositeXor : Compositor
    ∀ (a, b)
        >> a ⊕ b

CompositePlus : Compositor
    ∀ (a, b)
        >> (a + b) ◁ 1
```

```
CompositeOverlay : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
        >> {c if 2 × b < b.a, d}

CompositeDarken : Compositor
    ∀ (a, b)
        >> (a × b.a) ◁ (b × a.a) + (a ⊕ b)

CompositeLighten : Compositor
    ∀ (a, b)
        >> (a × b.a) ▷ (b × a.a) + (a ⊕ b)

CompositeColorDodge : Compositor
    ∀ (a, b)
        c = a.a × b.a + (a ⊕ b)
        d = (b × a.a / (1 - a / a.a) + (a ⊕ b)) ◁ 1
        >> {c if a × b.a + b × a.a ≥ a.a × b.a, d}

CompositeColorBurn : Compositor
    ∀ (a, b)
        c = a.a × (a × b.a + b × a.a - a.a × b.a) / a + (a ⊕ b)
        >> {a ⊕ b if a × b.a + b × a.a ≤ a.a × b.a, c}

    positeHardLight : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
    > {c if 2 × a < a.a, d}

    iteSoftLight : Compositor
    (a, b)
        c = (1 - b / b.a) × (2 × a - a.a)
        d = b × (a.a - c) + (a ⊕ b)
        e = b × (a.a - c × (3 - 8 × b / b.a)) + (a ⊕ b)
        f = b × a.a + (√(b / b.a) × b.a - b) × (2 × a - a.a) + (a ⊕ b)
        >> {d if 2 × a < a.a, e if b × 8 ≤ b.a, f}

CompositeDifference : Compositor
    ∀ (a, b)
        c = a + b - 2 × ((a × b.a) ◁ (b × a.a))
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeExclusion : Compositor
    ∀ (a, b)
        c = a × b.a + b × a.a - 2 × a × b + (a ⊕ b)
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeSubtract : Compositor
    ∀ (a, b)
        >> (a + b - 1) ▷ 0

CompositeInvert : Compositor
```

Nile Dataflow Language and Gezira Graphics Engine

Dan Amelang



Antialiased Vector Rendering

~ 45 LOC

Compositing

The 26 compositing ops
used by Flash/SVG,
Core Graphics, WPF

95 LOC

Pen Stroking

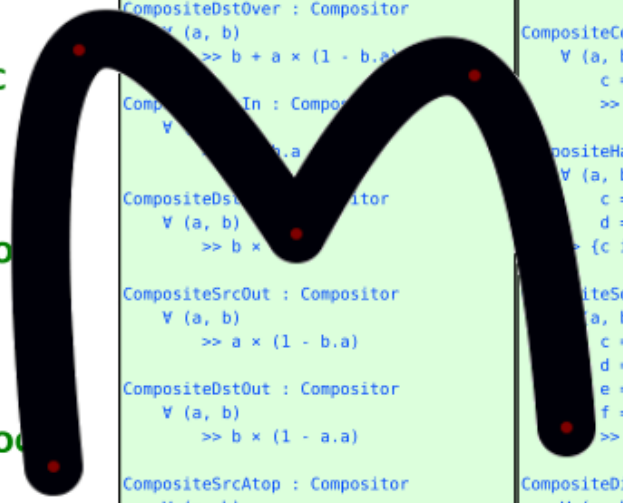
For outlines and drawings
3 kinds of caps and joins, et

95 LO

Texturing

For scaling, gradients,
Bit-map graphics, PNGs, etc.

71 LO

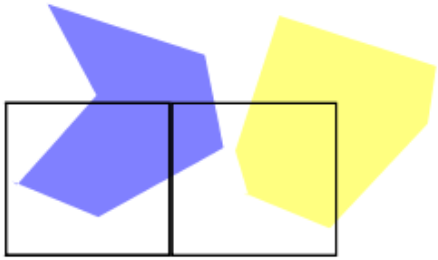


```
(a : Color) ⊕ (b : Color) : Color
    a × (1 - b.a) + b × (1 - a.a)

CompositeClear : Compositor
    ∀ (a, b)
        >> 0

CompositeSrc : Compositor
    ∀ (a, b)
        >> a

CompositeDst : Compositor
    ∀ (a, b)
        >> b

CompositeOver : Compositor
    ∀ (a, b)
        >> a + b × (1 - a.a)

CompositeDstOver : Compositor
    ∀ (a, b)
        >> b + a × (1 - b.a)

Comp        In : Compo
    ∀
            .a

CompositeDs        itor
    ∀ (a, b)
        >> b ×

CompositeSrcOut : Compositor
    ∀ (a, b)
        >> a × (1 - b.a)

CompositeDstOut : Compositor
    ∀ (a, b)
        >> b × (1 - a.a)

CompositeSrcAtop : Compositor
    ∀ (a, b)
        >> a × b.a + b × (1 - a.a)

CompositeDstAtop : Compositor
    ∀ (a, b)
        >> b × a.a + a × (1 - b.a)

CompositeXor : Compositor
    ∀ (a, b)
        >> a ⊕ b

CompositePlus : Compositor
    ∀ (a, b)
        >> (a + b) ◁ 1
```

```
CompositeOverlay : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
        >> {c if 2 × b < b.a, d}

CompositeDarken : Compositor
    ∀ (a, b)
        >> (a × b.a) ◁ (b × a.a) + (a ⊕ b)

CompositeLighten : Compositor
    ∀ (a, b)
        >> (a × b.a) ▷ (b × a.a) + (a ⊕ b)

CompositeColorDodge : Compositor
    ∀ (a, b)
        c = a.a × b.a + (a ⊕ b)
        d = (b × a.a / (1 - a / a.a) + (a ⊕ b)) ◁ 1
        >> {c if a × b.a + b × a.a ≥ a.a × b.a, d}

CompositeColorBurn : Compositor
    ∀ (a, b)
        c = a.a × (a × b.a + b × a.a - a.a × b.a) / a + (a ⊕ b)
        >> {a ⊕ b if a × b.a + b × a.a ≤ a.a × b.a, c}

    positeHardLight : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
    > {c if 2 × a < a.a, d}

    iteSoftLight : Compositor
    (a, b)
        c = (1 - b / b.a) × (2 × a - a.a)
        d = b × (a.a - c) + (a ⊕ b)
        e = b × (a.a - c × (3 - 8 × b / b.a)) + (a ⊕ b)
        f = b × a.a + (√(b / b.a) × b.a - b) × (2 × a - a.a) + (a ⊕ b)
        >> {d if 2 × a < a.a, e if b × 8 ≤ b.a, f}

CompositeDifference : Compositor
    ∀ (a, b)
        c = a + b - 2 × ((a × b.a) ◁ (b × a.a))
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeExclusion : Compositor
    ∀ (a, b)
        c = a × b.a + b × a.a - 2 × a × b + (a ⊕ b)
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeSubtract : Compositor
    ∀ (a, b)
        >> (a + b - 1) ▷ 0

CompositeInvert : Compositor
```

Nile Dataflow Language and Gezira Graphics Engine

Dan Amelang



Antialiased Vector Rendering

~ 45 LOC

Compositing

The 26 compositing ops used by Flash/SVG, Core Graphics, WPF

95 LOC

Pen Stroking

For outlines and drawings

3 kinds of caps and joins, et

95 LO

Texturing

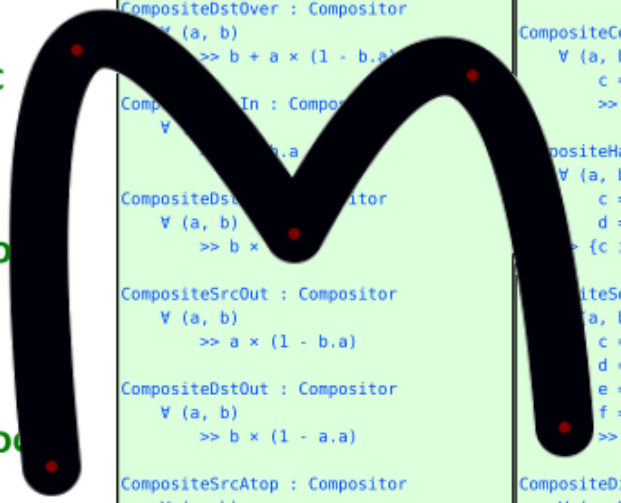
For scaling, gradients, Bit-map graphics, PNGs, etc.

71 LO

Bitmap Filters

Bilinear, bicubic, gaussian filters

129 LOC

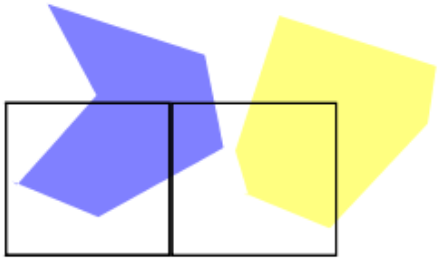


```
(a : Color) ⊕ (b : Color) : Color
    a × (1 - b.a) + b × (1 - a.a)

CompositeClear : Compositor
    ∀ (a, b)
        >> 0

CompositeSrc : Compositor
    ∀ (a, b)
        >> a

CompositeDst : Compositor
    ∀ (a, b)
        >> b

CompositeOver : Compositor
    ∀ (a, b)
        >> a + b × (1 - a.a)

CompositeDstOver : Compositor
    ∀ (a, b)
        >> b + a × (1 - b.a)

Comp        In : Compo
    ∀
            .a

CompositeDs        itor
    ∀ (a, b)
        >> b ×

CompositeSrcOut : Compositor
    ∀ (a, b)
        >> a × (1 - b.a)

CompositeDstOut : Compositor
    ∀ (a, b)
        >> b × (1 - a.a)

CompositeSrcAtop : Compositor
    ∀ (a, b)
        >> a × b.a + b × (1 - a.a)

CompositeDstAtop : Compositor
    ∀ (a, b)
        >> b × a.a + a × (1 - b.a)

CompositeXor : Compositor
    ∀ (a, b)
        >> a ⊕ b

CompositePlus : Compositor
    ∀ (a, b)
        >> (a + b) ◁ 1
```

```
CompositeOverlay : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
        >> {c if 2 × b < b.a, d}

CompositeDarken : Compositor
    ∀ (a, b)
        >> (a × b.a) ◁ (b × a.a) + (a ⊕ b)

CompositeLighten : Compositor
    ∀ (a, b)
        >> (a × b.a) ▷ (b × a.a) + (a ⊕ b)

CompositeColorDodge : Compositor
    ∀ (a, b)
        c = a.a × b.a + (a ⊕ b)
        d = (b × a.a / (1 - a / a.a) + (a ⊕ b)) ◁ 1
        >> {c if a × b.a + b × a.a ≥ a.a × b.a, d}

CompositeColorBurn : Compositor
    ∀ (a, b)
        c = a.a × (a × b.a + b × a.a - a.a × b.a) / a + (a ⊕ b)
        >> {a ⊕ b if a × b.a + b × a.a ≤ a.a × b.a, c}

    positeHardLight : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
    > {c if 2 × a < a.a, d}

    iteSoftLight : Compositor
    (a, b)
        c = (1 - b / b.a) × (2 × a - a.a)
        d = b × (a.a - c) + (a ⊕ b)
        e = b × (a.a - c × (3 - 8 × b / b.a)) + (a ⊕ b)
        f = b × a.a + (√(b / b.a) × b.a - b) × (2 × a - a.a) + (a ⊕ b)
        >> {d if 2 × a < a.a, e if b × 8 ≤ b.a, f}

CompositeDifference : Compositor
    ∀ (a, b)
        c = a + b - 2 × ((a × b.a) ◁ (b × a.a))
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeExclusion : Compositor
    ∀ (a, b)
        c = a × b.a + b × a.a - 2 × a × b + (a ⊕ b)
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeSubtract : Compositor
    ∀ (a, b)
        >> (a + b - 1) ▷ 0

CompositeInvert : Compositor
```

Nile Dataflow Language and Gezira Graphics Engine

Dan Amelang



Antialiased Vector Rendering

~ 45 LOC

Compositing

The 26 compositing ops
used by Flash/SVG,
Core Graphics, WPF

95 LOC

Pen Stroking

For outlines and drawings
3 kinds of caps and joins, et

95 LO

Texturing

For scaling, gradients,
Bit-map graphics, PNGs, etc.

71 LO

Bitmap Filters

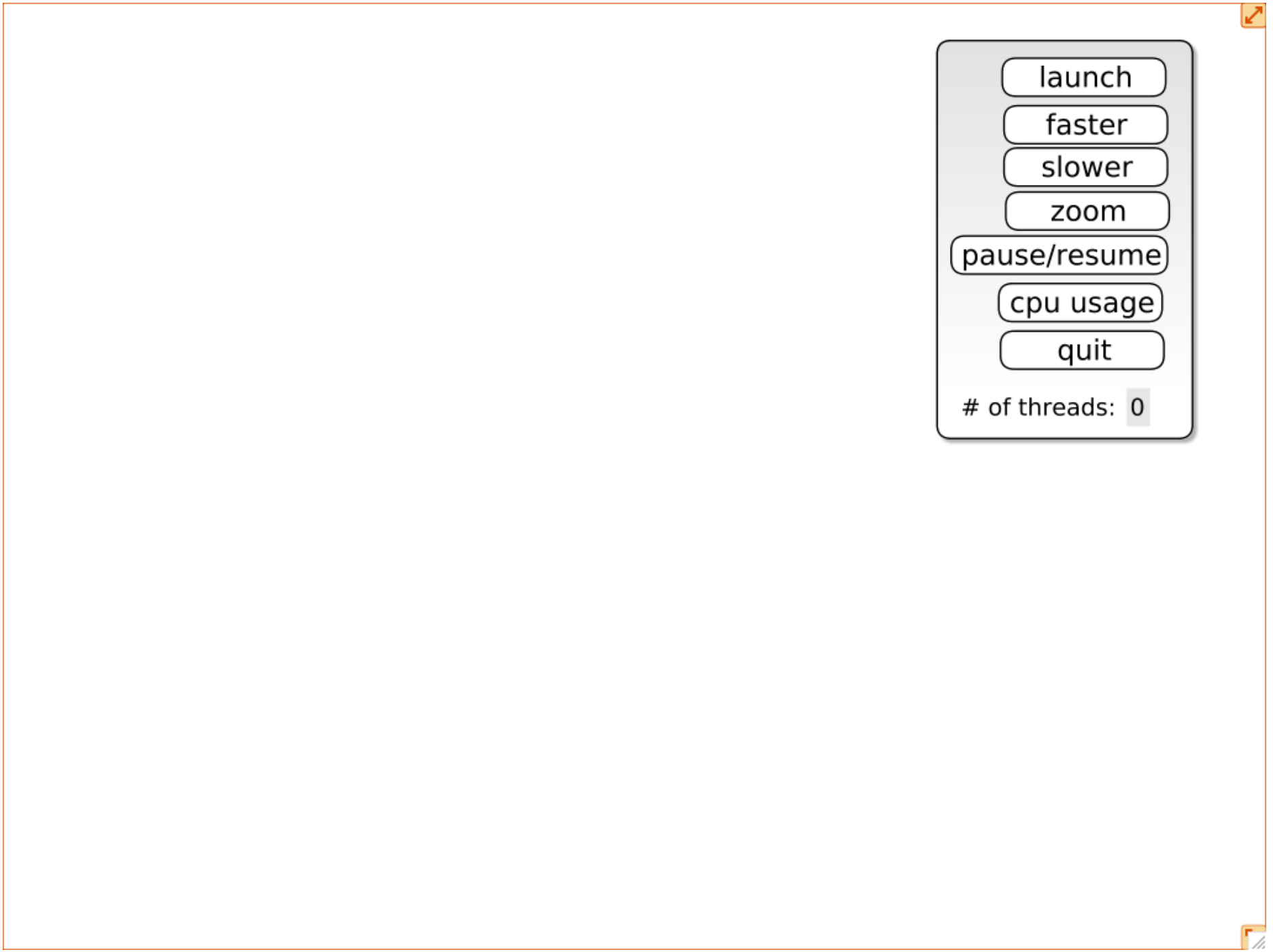
Bilinear, bicubic, gaussian filters

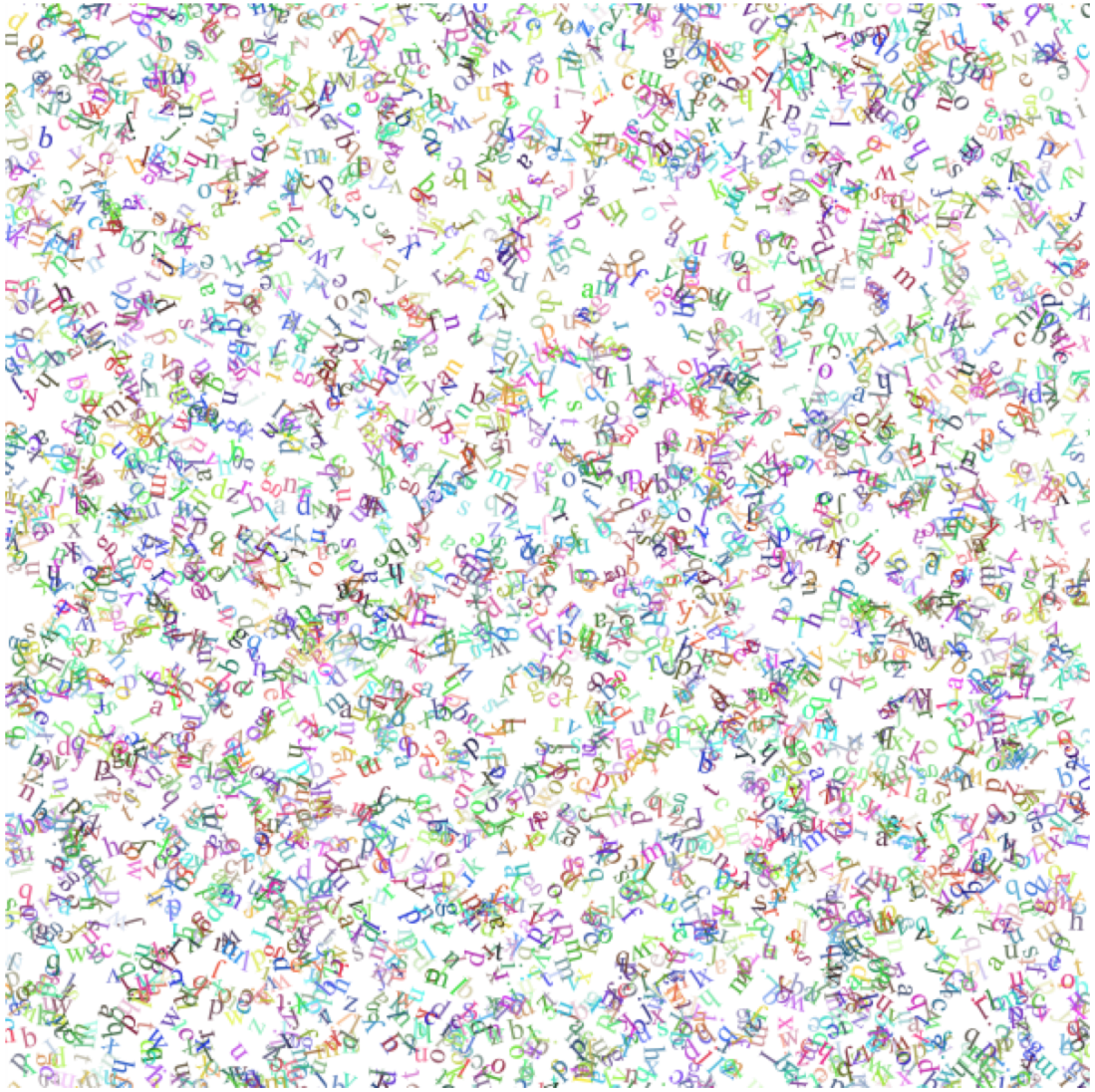
129 LOC

Nile Total 435 LOC

```
(a : Color) ⊕ (b : Color) : Color
    a × (1 - b.a) + b × (1 - a.a)

CompositeClear : Compositor
    ∀ (a, b)
        >> 0

CompositeSrc : Compositor
    ∀ (a, b)
        >> a

CompositeDst : Compositor
    ∀ (a, b)
        >> b

CompositeOver : Compositor
    ∀ (a, b)
        >> a + b × (1 - a.a)

CompositeDstOver : Compositor
    ∀ (a, b)
        >> b + a × (1 - b.a)

Comp        In : Compo
    ∀
            .a

CompositeDs        itor
    ∀ (a, b)
        >> b ×

CompositeSrcOut : Compositor
    ∀ (a, b)
        >> a × (1 - b.a)

CompositeDstOut : Compositor
    ∀ (a, b)
        >> b × (1 - a.a)

CompositeSrcAtop : Compositor
    ∀ (a, b)
        >> a × b.a + b × (1 - a.a)

CompositeDstAtop : Compositor
    ∀ (a, b)
        >> b × a.a + a × (1 - b.a)

CompositeXor : Compositor
    ∀ (a, b)
        >> a ⊕ b

CompositePlus : Compositor
    ∀ (a, b)
        >> (a + b) ◁ 1
```

```
CompositeOverlay : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
        >> {c if 2 × b < b.a, d}

CompositeDarken : Compositor
    ∀ (a, b)
        >> (a × b.a) ◁ (b × a.a) + (a ⊕ b)

CompositeLighten : Compositor
    ∀ (a, b)
        >> (a × b.a) ▷ (b × a.a) + (a ⊕ b)

CompositeColorDodge : Compositor
    ∀ (a, b)
        c = a.a × b.a + (a ⊕ b)
        d = (b × a.a / (1 - a / a.a) + (a ⊕ b)) ◁ 1
        >> {c if a × b.a + b × a.a ≥ a.a × b.a, d}

CompositeColorBurn : Compositor
    ∀ (a, b)
        c = a.a × (a × b.a + b × a.a - a.a × b.a) / a + (a ⊕ b)
        >> {a ⊕ b if a × b.a + b × a.a ≤ a.a × b.a, c}

positeHardLight : Compositor
    ∀ (a, b)
        c = 2 × a × b + (a ⊕ b)
        d = a.a × b.a - 2 × (b.a - b) × (a.a - a) + (a ⊕ b)
    > {c if 2 × a < a.a, d}

iteSoftLight : Compositor
    a, b)
        c = (1 - b / b.a) × (2 × a - a.a)
        d = b × (a.a - c) + (a ⊕ b)
        e = b × (a.a - c × (3 - 8 × b / b.a)) + (a ⊕ b)
        f = b × a.a + (√(b / b.a) × b.a - b) × (2 × a - a.a) + (a ⊕ b)
        >> {d if 2 × a < a.a, e if b × 8 ≤ b.a, f}

CompositeDifference : Compositor
    ∀ (a, b)
        c = a + b - 2 × ((a × b.a) ◁ (b × a.a))
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeExclusion : Compositor
    ∀ (a, b)
        c = a × b.a + b × a.a - 2 × a × b + (a ⊕ b)
        >> (c.a + a.a × b.a, c.r, c.g, c.b)

CompositeSubtract : Compositor
    ∀ (a, b)
        >> (a + b - 1) ▷ 0

CompositeInvert : Compositor
```


launch
faster
slower
zoom
pause/resume
cpu usage
quit
# of threads: 0



launch

faster

slower

zoom

pause/resume

cpu usage

quit

# of threads: 1

Bret Victor

We need to continuously experience the Meanings of our Meanings

Nile Viewer

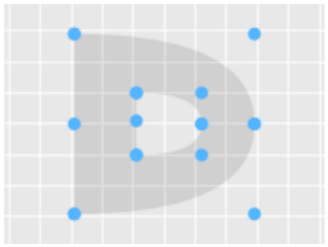
Nile Viewer

file:///Users/ohshima/src/Squeak/Frank-120427/high_contrast.html

Google

initial input

6 Beziers

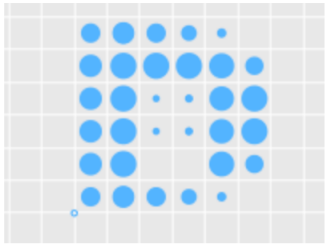


Bret Victor

▶ Rasterize

processed 6 Beziers

output 49 SpanCoverages

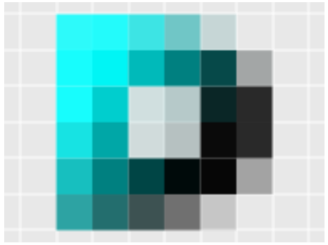


```
Rasterize () : Bezier >> EdgeSpan
    → DecomposeBeziers () → SortBy (1) → SortBy (2) → CombineEdgeSamples ()
```

▶ Texture

processed 49 SpanCoverages

output 34 Pixels



```
Texture (A:ColorStop, B:ColorStop) : EdgeSpan >> (Color, PointCoverage)
    → ExpandSpans () → DupZip (→ ProjectLinearGradient (A.P, B.P) -> PadGradient () -> GradientSpan (A.C, B.C),
                               → PassThrough ())
```

Nile Viewer

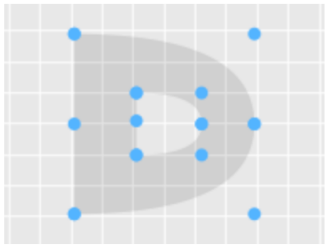
Nile Viewer

file:///Users/ohshima/src/Squeak/Frank-120427/high_contrast.html

Google

initial input

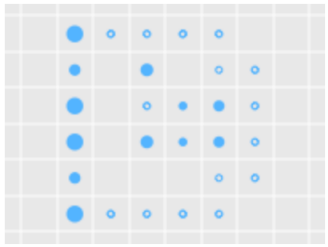
6 Beziers



DecomposeBeziers

processed 114 Beziers

output 60 EdgeSamples



```
DecomposeBeziers () : Bezier >> EdgeSample
    ϵ = 0.1
    ∀ (A, B, C)
        inside = (⌊ A ⌋ = ⌊ C ⌋ ∨ ⌈ A ⌉ =
        if inside.x ∧ inside.y
            P = ⌊A⌋ ◁ ⌊C⌋
            (x, y) = P
            (w, _) = P + 1 - (A ~ C)
            (_, h) = C - A
            >> (x + 0.5, y + 0.5, wh, h)
        else
            M           = (A ~ B) ~ (B ~ C
            ( min,  max) = (⌊M⌋, ⌈M⌉)
            (Δmin, Δmax) = (M - min, M -
            N = { min, if |Δmin| < ϵ
                  max, if |Δmax| < ϵ
                    M,     otherwise }
            << (N, B ~ C, C) << (A, A ~ B,
```

▼ **Rasterize**

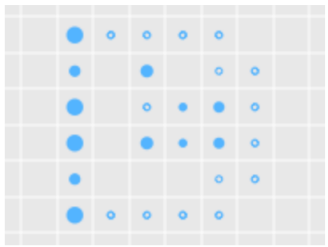
processed 6 Beziers

output 49 SpanCoverages

Sort

processed 60 EdgeSamples

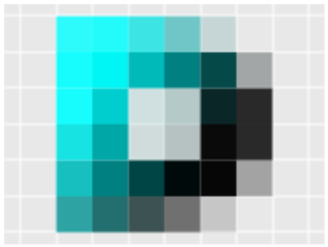
output 60 EdgeSamples



▶ **Texture**

processed 49 SpanCoverages

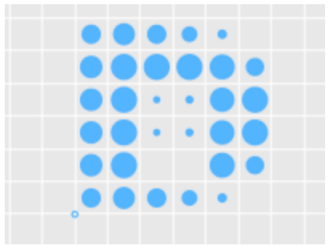
output 34 Pixels



CombineEdgeSamples

processed 60 EdgeSamples

output 49 SpanCoverages



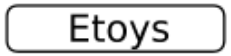
```
CombineEdgeSamples () : EdgeSample >> SpanCo
    (x, y, A, H) = (0, 0, 0, 0)
    ∀ (x', y', a, h)
        if y' = y
            if x' = x
                (A', H') = (A + a, H + h)
            else
                (A', H') = (H + a, H + h)
                >> (x,     y, |A| ◁ 1,
                >> (x + 1, y, |H| ◁ 1, x'
        else
            (A', H') = (a, h)
            >> (x, y, |A| ◁ 1, 1)
    >> (x, y, |A| ◁ 1, 1)
```

An App Framework

What do we want in a GUI framework?

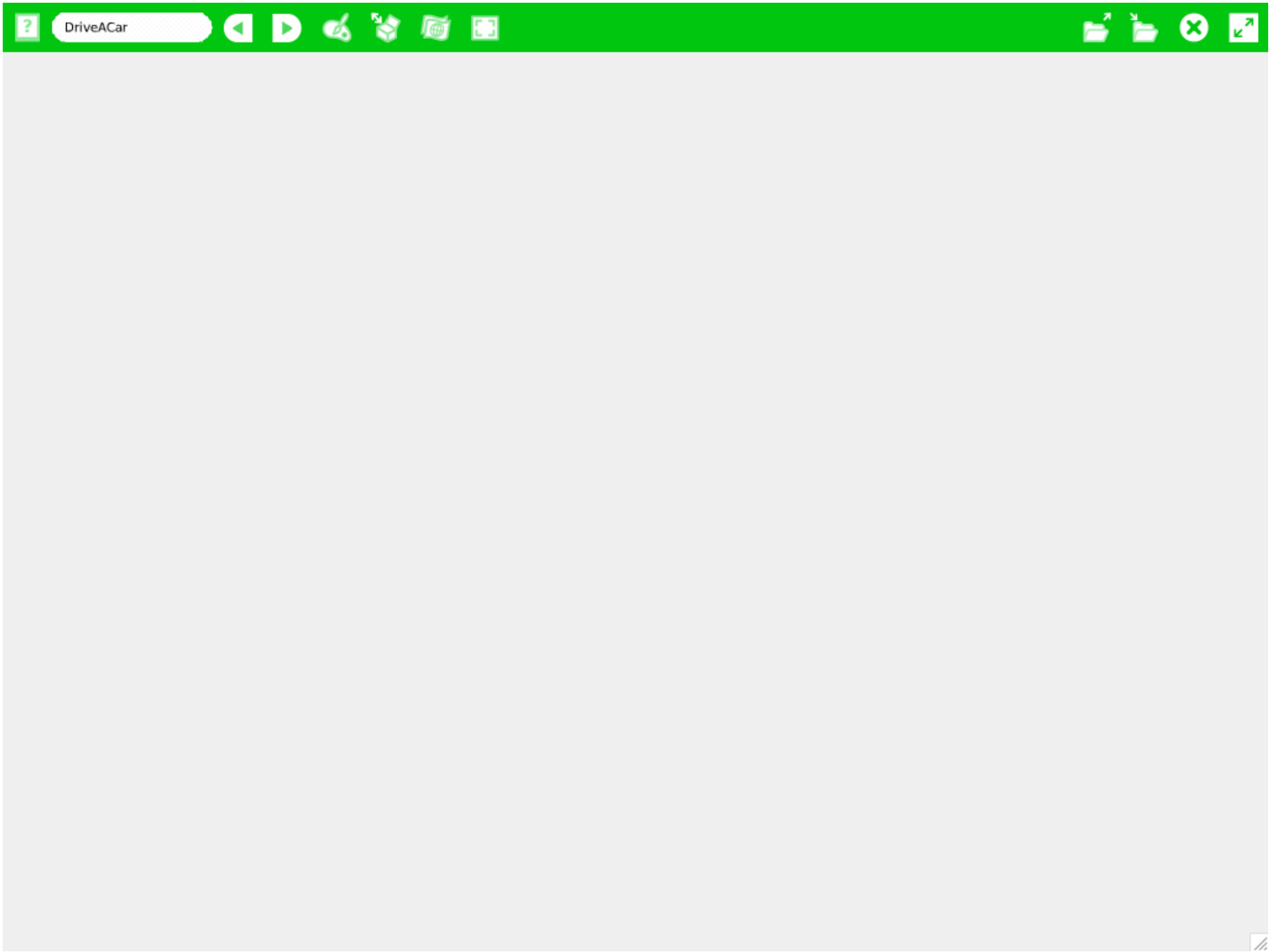
A uniform graphical object model.

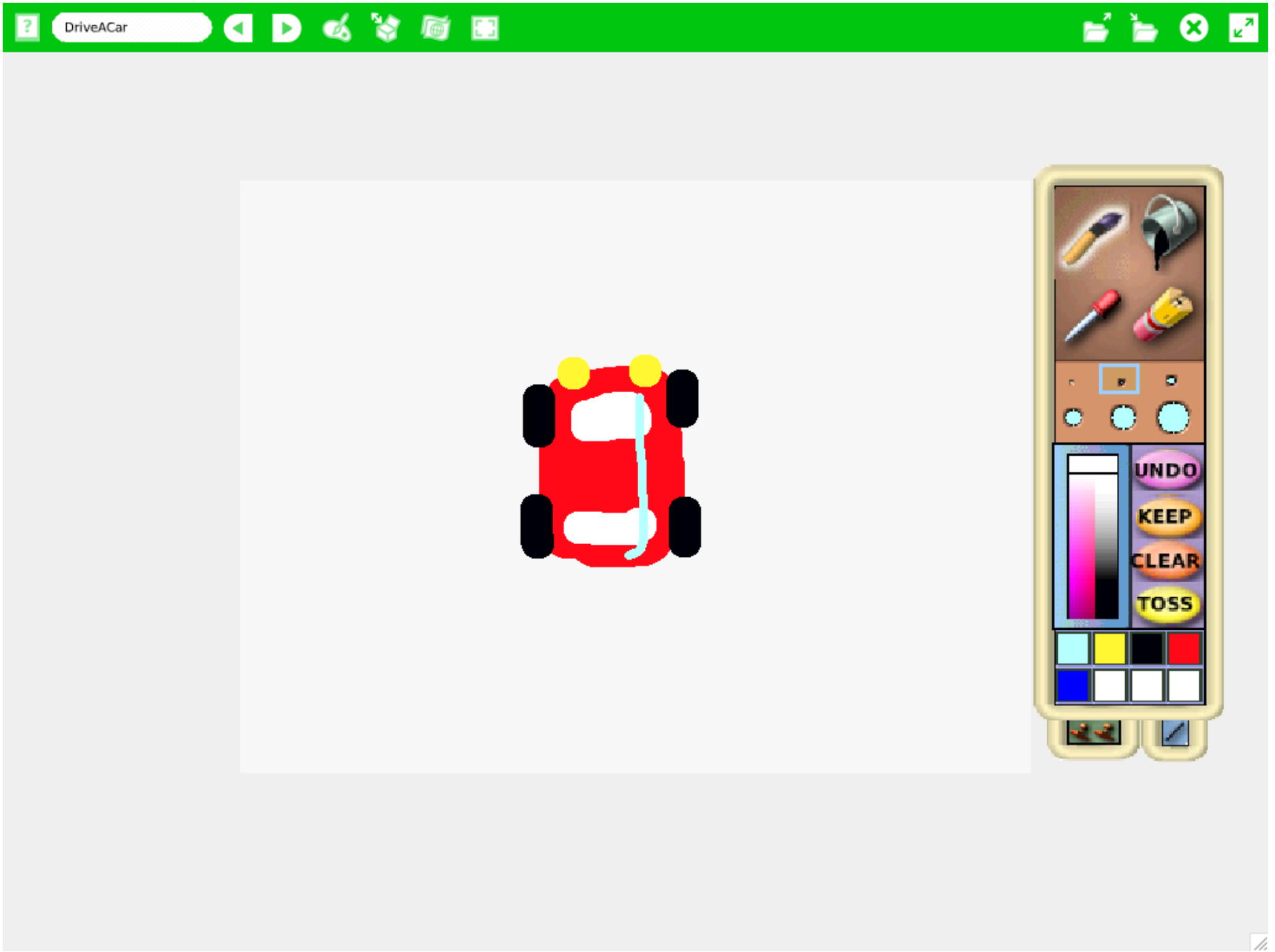
We at least want to do the stuff we can do in Etoys.



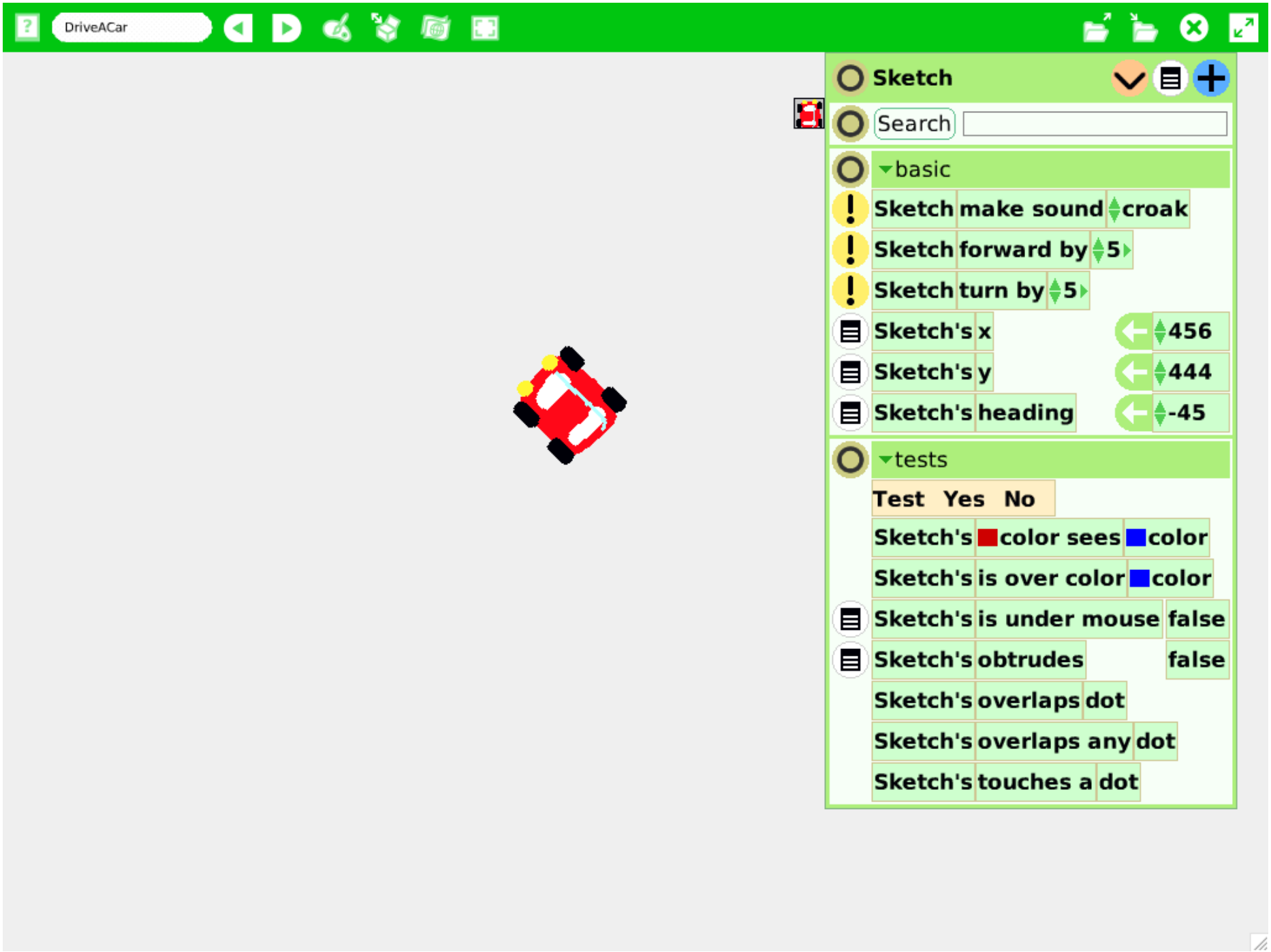
Can we have a clean, time-aware implementation?

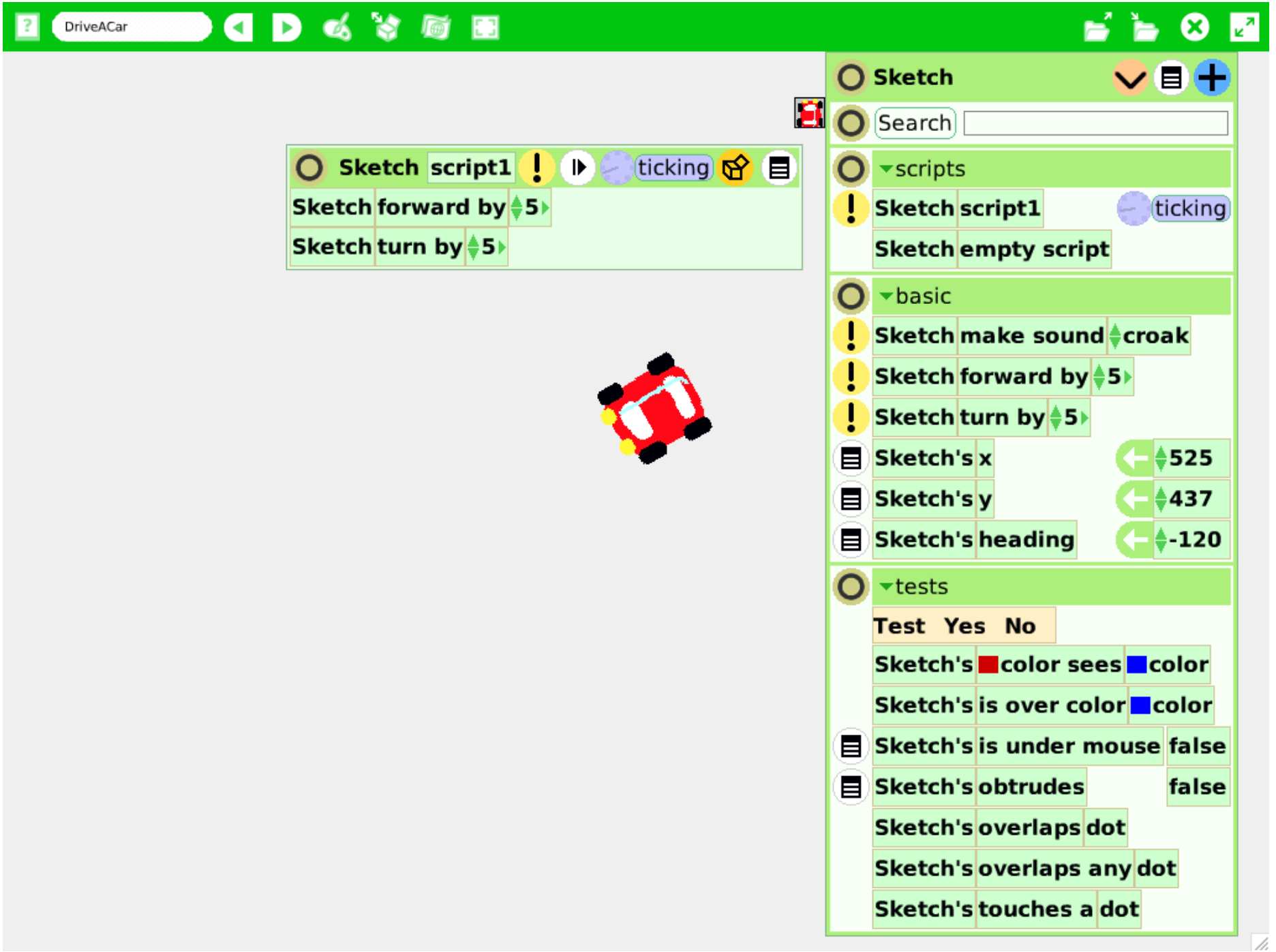
Functional Reactive Programming (FRP) seems to be a good way.

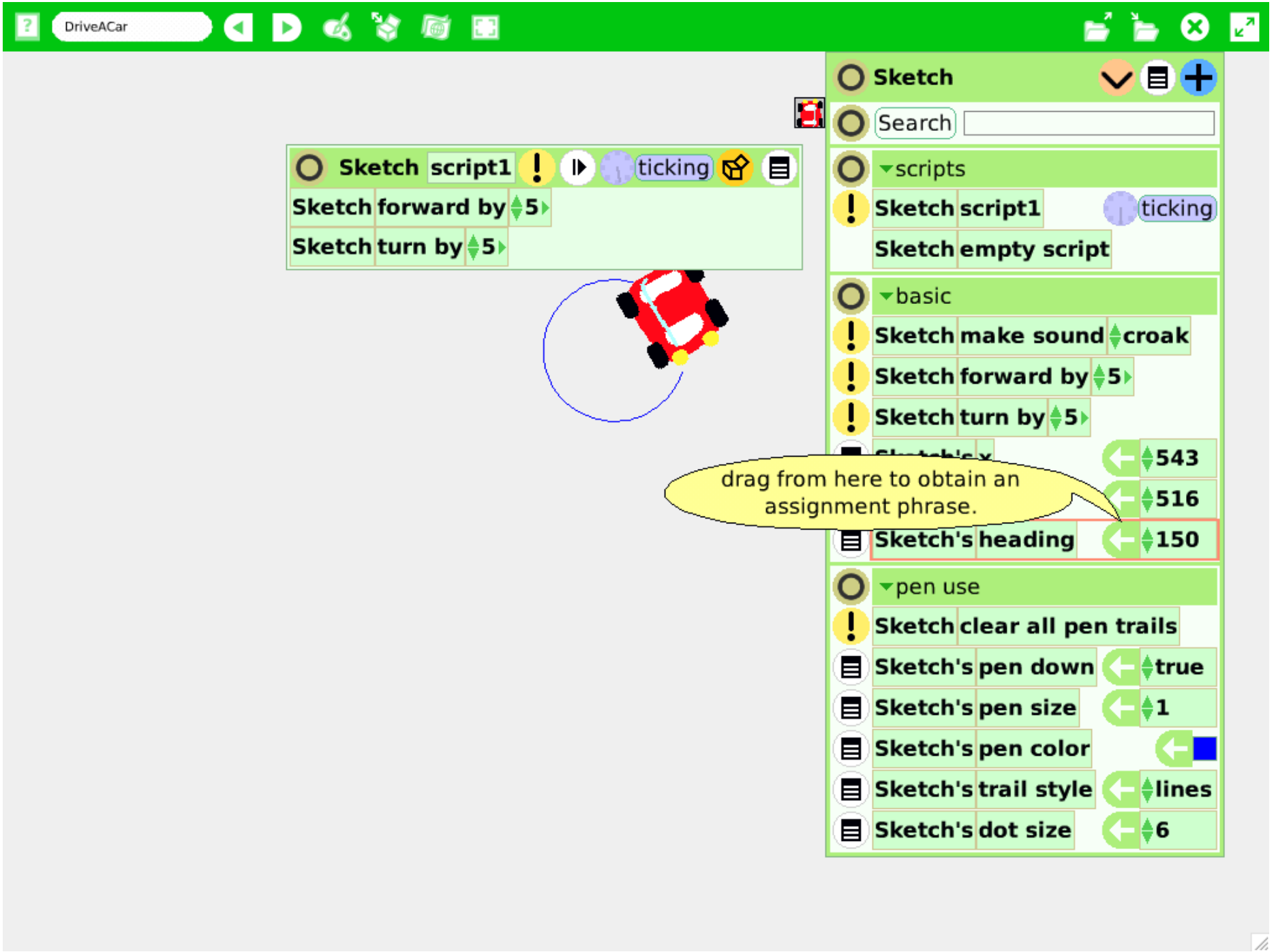

DriveACar

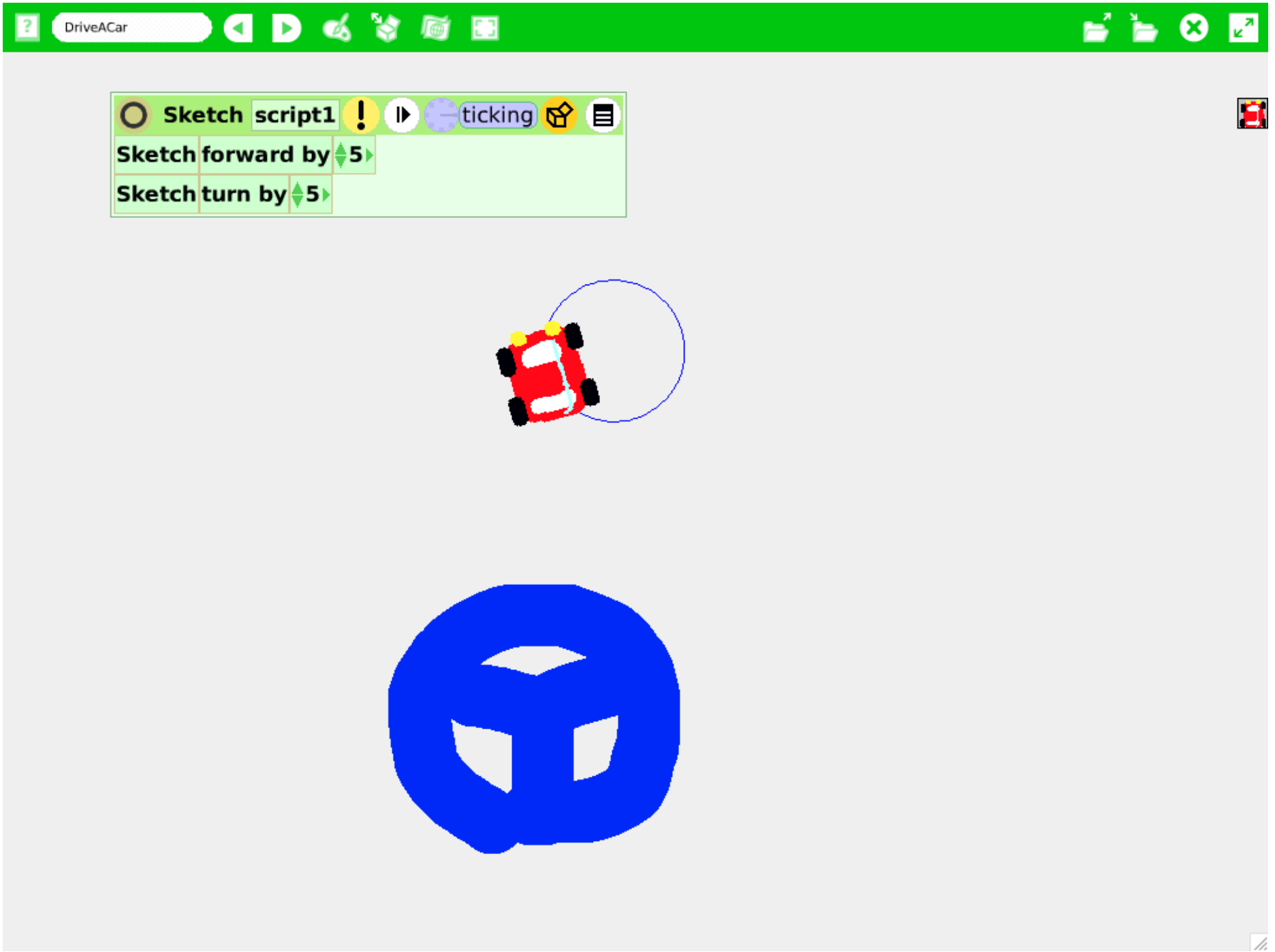

DriveACar
UNDO
KEEP
CLEAR
TOSS

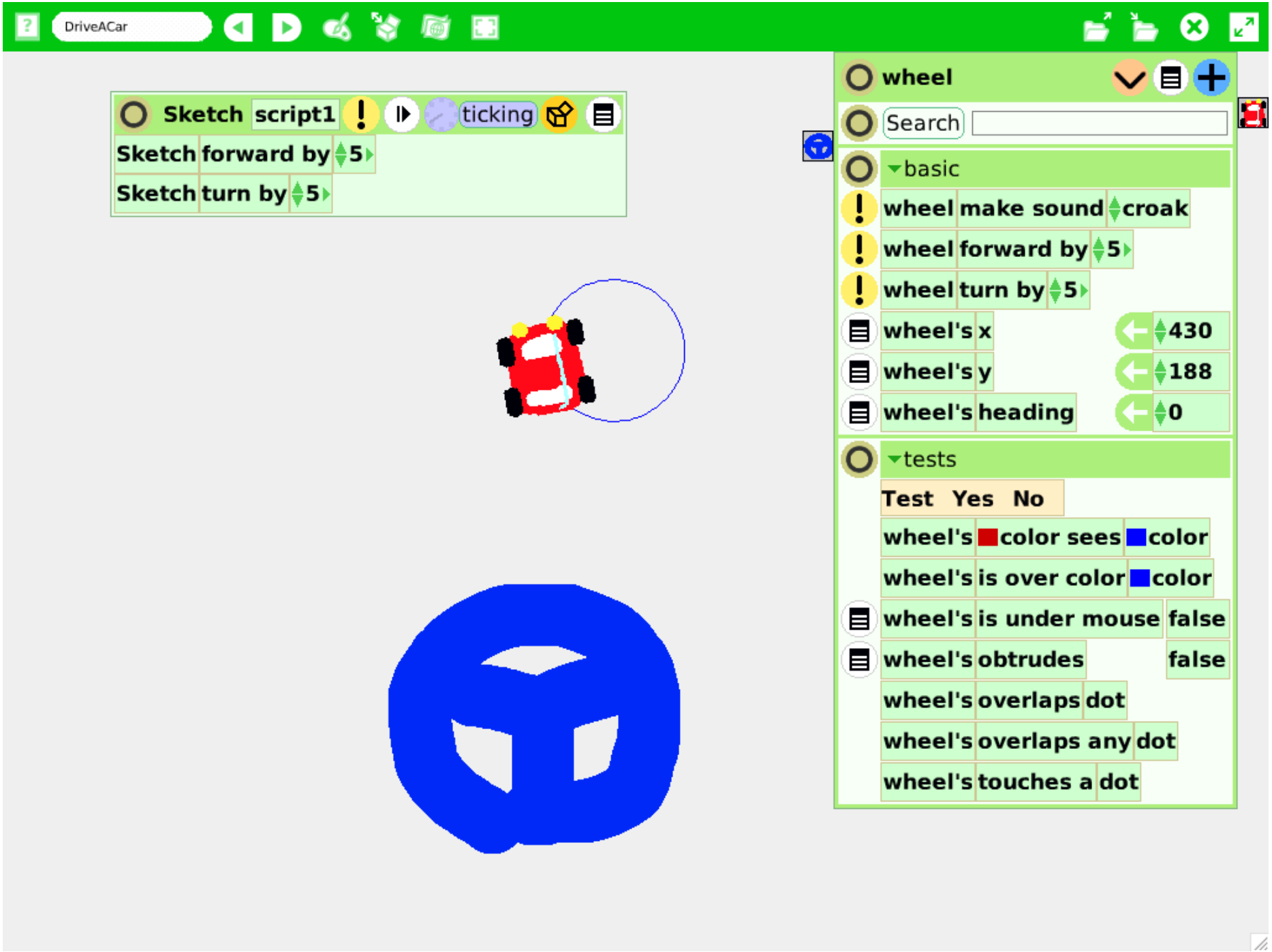

Sketch

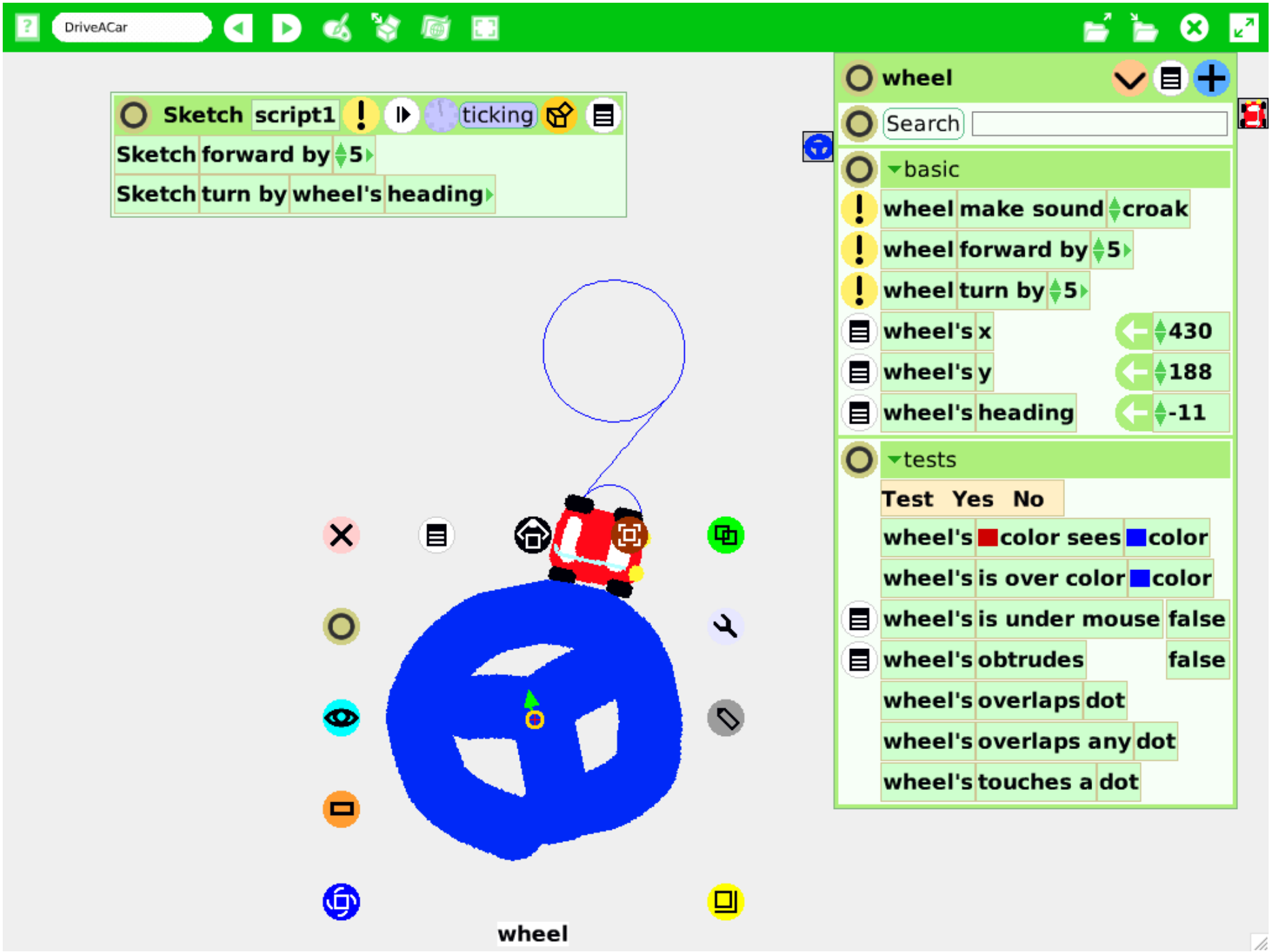

DriveACar
Sketch
Search
basic
Sketch make sound croak
Sketch forward by 5
Sketch turn by 5
Sketch's x 456
Sketch's y 444
Sketch's heading -45
tests
Test Yes No
Sketch's color sees color
Sketch's is over color color
Sketch's is under mouse false
Sketch's obtrudes false
Sketch's overlaps dot
Sketch's overlaps any dot
Sketch's touches a dot

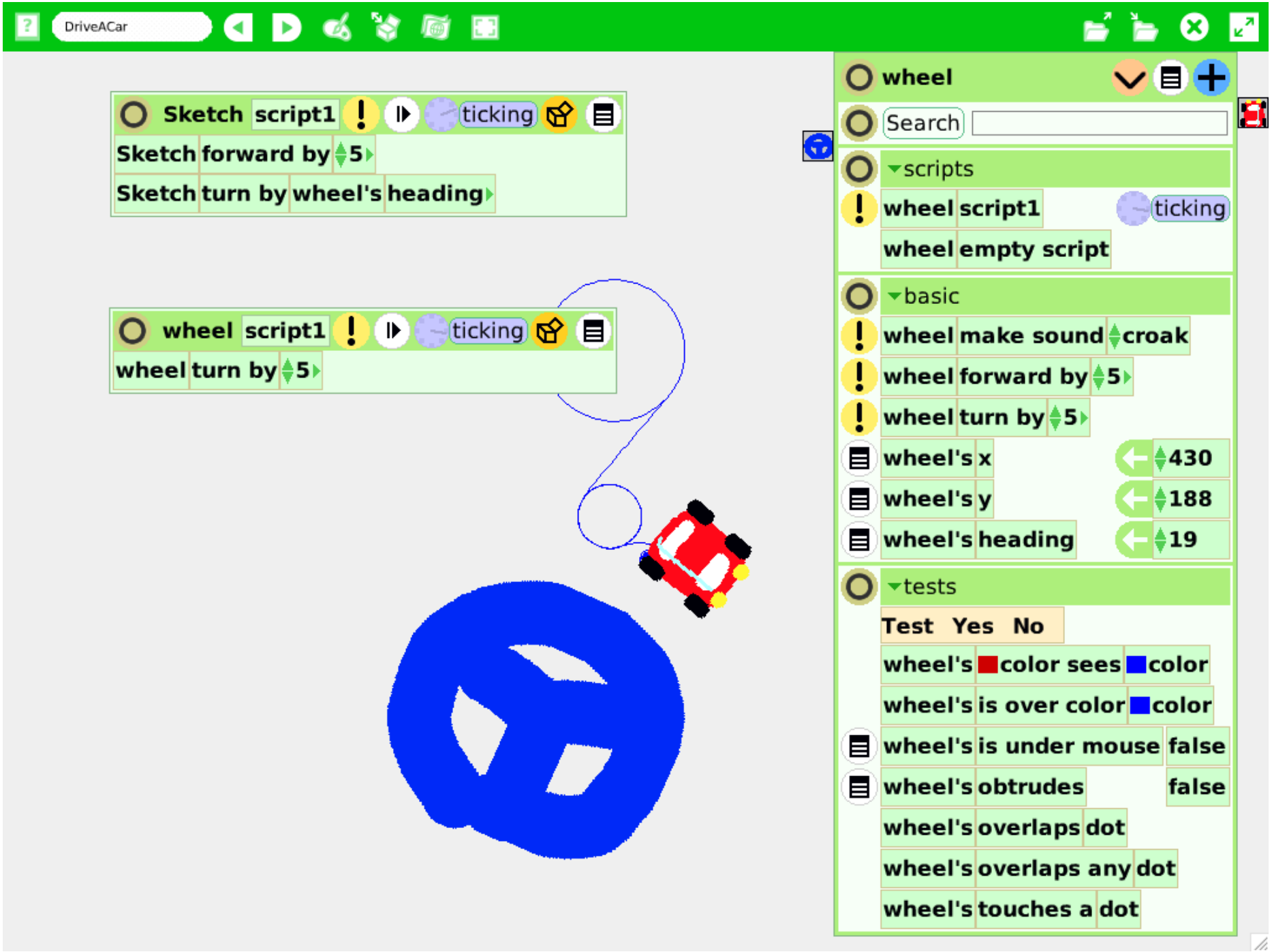

DriveACar
Sketch
Search
scripts
Sketch script1
ticking
Sketch empty script
basic
Sketch make sound croak
Sketch forward by 5
Sketch turn by 5
Sketch's x 525
Sketch's y 437
Sketch's heading -120
tests
Test Yes No
Sketch's color sees color
Sketch's is over color color
Sketch's is under mouse false
Sketch's obtrudes false
Sketch's overlaps dot
Sketch's overlaps any dot
Sketch's touches a dot
Sketch script1 ticking
Sketch forward by 5
Sketch turn by 5

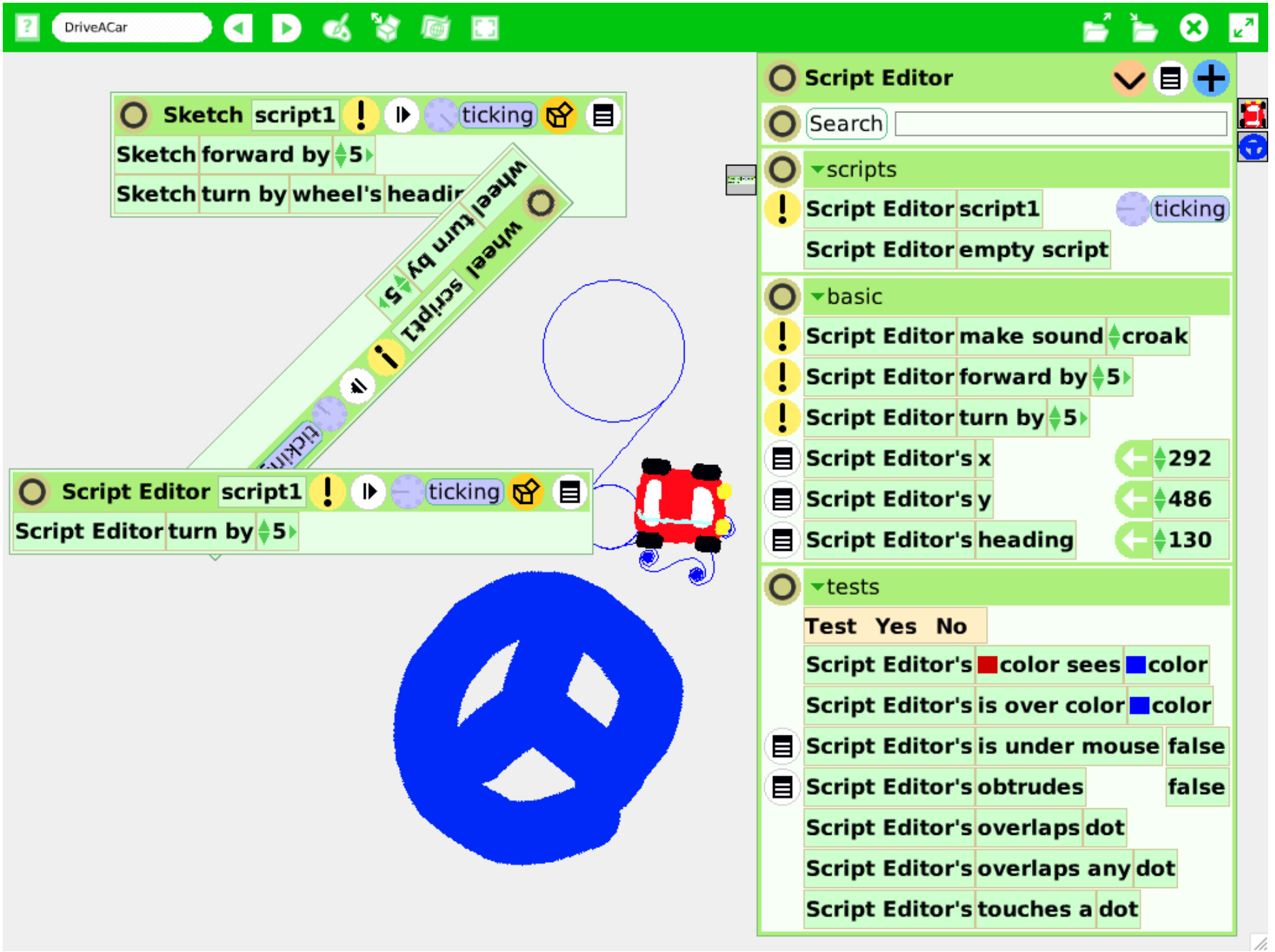

DriveACar
Sketch
Search
scripts
Sketch script1
ticking
Sketch empty script
basic
Sketch make sound croak
Sketch forward by 5
Sketch turn by 5
543
516
Sketch's heading 150
drag from here to obtain an assignment phrase.
pen use
Sketch clear all pen trails
Sketch's pen down true
Sketch's pen size 1
Sketch's pen color
Sketch's trail style lines
Sketch's dot size 6
Sketch script1 ticking
Sketch forward by 5
Sketch turn by 5


DriveACar
Sketch script1
ticking
Sketch forward by 5
Sketch turn by 5


DriveACar
Sketch script1 ticking
Sketch forward by 5
Sketch turn by 5
wheel
Search
basic
wheel make sound croak
wheel forward by 5
wheel turn by 5
wheel's x 430
wheel's y 188
wheel's heading 0
tests
Test Yes No
wheel's color sees color
wheel's is over color color
wheel's is under mouse false
wheel's obtrudes false
wheel's overlaps dot
wheel's overlaps any dot
wheel's touches a dot


DriveACar
Sketch script1 ticking
Sketch forward by 5
Sketch turn by wheel's heading
wheel
Search
basic
wheel make sound croak
wheel forward by 5
wheel turn by 5
wheel's x 430
wheel's y 188
wheel's heading -11
tests
Test Yes No
wheel's color sees color
wheel's is over color color
wheel's is under mouse false
wheel's obtrudes false
wheel's overlaps dot
wheel's overlaps any dot
wheel's touches a dot
wheel


DriveACar
Sketch script1 ticking
Sketch forward by 5
Sketch turn by wheel's heading
wheel script1 ticking
wheel turn by 5
wheel
Search
scripts
wheel script1 ticking
wheel empty script
basic
wheel make sound croak
wheel forward by 5
wheel turn by 5
wheel's x 430
wheel's y 188
wheel's heading 19
tests
Test Yes No
wheel's color sees color
wheel's is over color color
wheel's is under mouse false
wheel's obtrudes false
wheel's overlaps dot
wheel's overlaps any dot
wheel's touches a dot


DriveACar
Sketch script1 ticking
Sketch forward by 5
Sketch turn by wheel's headi
wheel turn by 5
wheel script1 i
Script Editor script1 ticking
Script Editor turn by 5
Script Editor
Search
scripts
Script Editor script1 ticking
Script Editor empty script
basic
Script Editor make sound croak
Script Editor forward by 5
Script Editor turn by 5
Script Editor's x 292
Script Editor's y 486
Script Editor's heading 130
tests
Test Yes No
Script Editor's color sees color
Script Editor's is over color color
Script Editor's is under mouse false
Script Editor's obtrudes false
Script Editor's overlaps dot
Script Editor's overlaps any dot
Script Editor's touches a dot

GUI framework for exploration

- Supporting Differential/Turtle Geometry

car turn by 5

car forward by 10

GUI framework for exploration

- Supporting Differential/Turtle Geometry

```
car turn by 5
car forward by 10
```

- Incremental Development

GUI framework for exploration

- Supporting Differential/Turtle Geometry

car turn by 5

car forward by 10

- Incremental Development

car turn by 5

car forward by 10

GUI framework for exploration

- Supporting Differential/Turtle Geometry

car turn by 5

car forward by 10

- Incremental Development

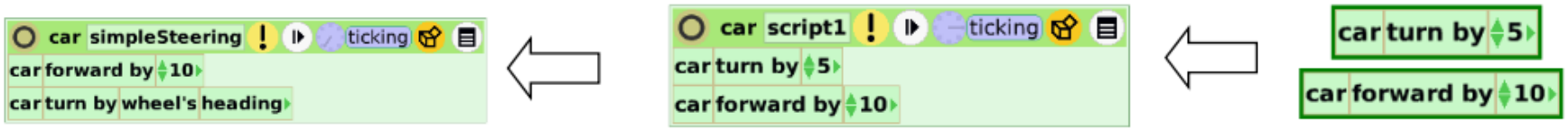
car turn by 5

car forward by 10

GUI framework for exploration

- Supporting Differential/Turtle Geometry

- Incremental Development



GUI framework for exploration

- Supporting Differential/Turtle Geometry

car turn by 5

car forward by 10

- Incremental Development

car simpleSteering ticking

car forward by 10

car turn by wheel's heading

car script1 ticking

car turn by 5

car forward by 10

car turn by 5

car forward by 10

car turn by wheel's heading

GUI framework for exploration

- Supporting Differential/Turtle Geometry

car turn by 5

car forward by 10

- Incremental Development

car simpleSteering ticking

car forward by 10

car turn by wheel's heading

car script1 ticking

car turn by 5

car forward by 10

car turn by 5

car forward by 10

car turn by wheel's heading

FrTime:

```
(define uchrs (map-e char-upcase chrs))
(define clst (collect-e uchrs empty cons))
(list->string (reverse (hold clst empty)))
```

KScript

An OO language with Functional Reactive Programming (FRP)-inspired extensions.

KScript

An OO language with Functional Reactive Programming (FRP)-inspired extensions.

- variable-sized key-value objects with methods.

KScript

An OO language with Functional Reactive Programming (FRP)-inspired extensions.

- variable-sized key-value objects with methods.

- FRP extension provides a declarative way to specify dependencies.

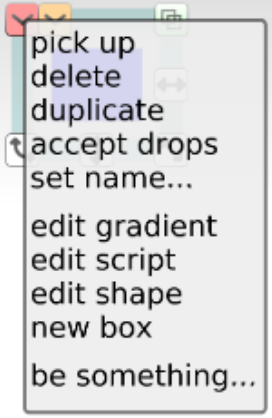
KScript

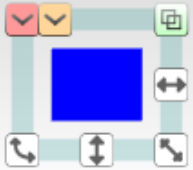
An OO language with Functional Reactive Programming (FRP)-inspired extensions.

- variable-sized key-value objects with methods.

- FRP extension provides a declarative way to specify dependencies.

- But keep objects loosely-coupled.


pick up
delete
duplicate
accept drops
set name...
edit gradient
edit script
edit shape
new box
be something...



aBox

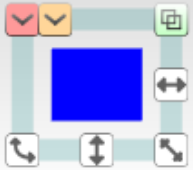
+/-

b

b <- a + 1

b = undefined

b <- a + 1



aBox

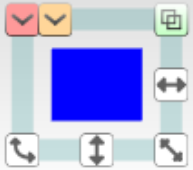
+/-

a
b

a <- 10
b <- a + 1

a = 10
b = 11

b <- a + 1
a <- 10



aBox

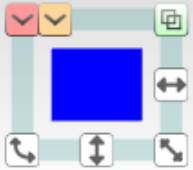
+/-

a
b
c

```
a <- 10
b <- a + 1
c <- a * b
```

```
a = 10
b = 11
c = 110
```

```
b <- a + 1
a <- 10
c <- a * b
```



aBox

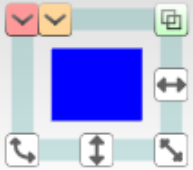
+/-

timerE(1000)

a
b
c

75000

```
b <- a + 1
a <- 10
c <- a * b
```



aBox

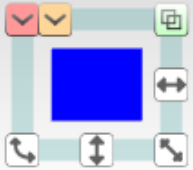
+/-

a
b
c

```
a <- timerE(1000)
b <- a + 1
c <- a * b
```

```
a = 81000
b = 81001
c = 6561081000
```

```
b <- a + 1
a <- 10
c <- a * b
```



aBox

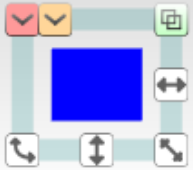
+/-

b
c

b <- a + 1
c <- a * b

b = 94001
c = 8836094000

delete this.a



aBox

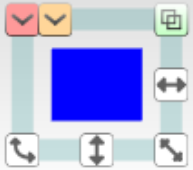
+/-

undefined

b
buttonDown
c

undefined

buttonDown <- undefined



aBox

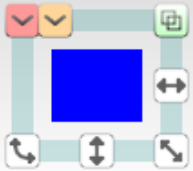
+/-

undefined

b
buttonDown
c

UserEvent {buttons: 4,
defaultHandler: ..., event:
"buttonDown", hand: ..., handler: ...,
isMouseEvent: true, position:
P(227,174), root: ..., time: 118179,
timeStamp: 5123817}

buttonDown <- undefined



aBox

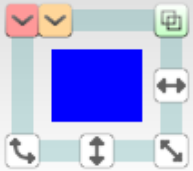
+/-

undefined

b
buttonDown
c
mover

UserEvent {buttons: 4, defaultHandler: ..., event: "buttonDown", hand: ..., handler: ..., isMouseEvent: true, position: P(227,174), root: ..., time: 118179, timeStamp: 5123817}

```
mover <- this.center(buttonDown.position)
```



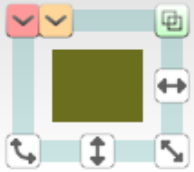
aBox

+/-

M(1.0, 0.0, 0.0, 0.0, 1.0, 0.0)

c
clipping
container
contents
desiredExtent
extent
glVertices
layout
layoutChanged
mover
name
optimizedField
optimizedGlyph
parts
pivotRatio
shape
transformation
visible

M(1.0, 0.0, 202.0, 0.0, 1.0, 154.0)



aBox

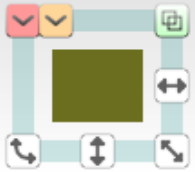
+/-

b
buttonDown
c
colorChanger
mover

```
b <- a + 1
buttonDown <- undefined
c <- a * b
colorChanger <- when buttonDown
then this.fill(:Color.random())
mover <-
this.center(buttonDown.position)
```

```
b = 94001
buttonDown =
KSUserEvent(#buttons->4,
#defaultHandler->...,
#event->buttonDown, #hand->...,
#handler->..., #isMouseEvent->true,
#position->218@189, #root->...,
```

```
colorChanger <- when buttonDown then
this.fill(:Color.random())
```



aBox

+/-

b
buttonDown
c
mover

```
b <- a + 1
buttonDown <- undefined
c <- a * b
mover <-
this.center(buttonDown.position)
```

```
b = 94001
buttonDown =
KSUserEvent(#buttons->4,
#defaultHandler->...,
#event->buttonDown, #hand->...,
#handler->..., #isMouseEvent->true,
#position->218@189, #root->...,
```

```
delete this.colorChanger
```

Kscript: Evaluation

All fields of an object are reactive variables.

Kscript: Evaluation

All fields of an object are reactive variables.

At every display cycle:

Kscript: Evaluation

All fields of an object are reactive variables.

At every display cycle:

Topologically sort variables.

Kscript: Evaluation

All fields of an object are reactive variables.

At every display cycle:

Topologically sort variables.

Evaluate variables to be updated.

Loose-Coupled Formulae

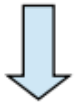
formula	current value
b <- a + 1	101
a <- timerE(100)	100

b <- a + 1

Loose-Coupled Formulae

formula	current value
b <- a + 1	101
a <- timerE(100)	100

b <- a + 1

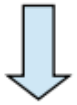


this.b = new EventStream((x) -> x + 1, ["a"])

Loose-Coupled Formulae

formula	current value
b <- a + 1	101
a <- timerE(100)	100

b <- a + 1



this.b = new EventStream((x) -> x + 1, ["a"])

(A variable does not contain any pointers to others.)

Everything can be extended from outside!

pressed <- ... buttonDown ... buttonUp ...

Everything can be extended from outside!

pressed <- … buttonDown … buttonUp …

doAction <- when mergeE(pressed, …) then …

Everything can be extended from outside!

```
pressed <- … buttonDown … buttonUp …

doAction <- when mergeE(pressed, …) then …

doSomethingElse <- when
      pressed
   then …
```

Everything can be extended from outside!

```
pressed <- ... buttonDown ... buttonUp ...

doAction <- when mergeE(pressed, ...) then ...

doSomethingElse <- when
      pressed
   then ...
```

```
onPressed: anEvent
   self doAction.
   self doSomethingElse.
```

Don't cry over a dropped card deck!

Ellis D. Kropotechev and Zeus,
A Marvelous Time-Sharing Device ('67)

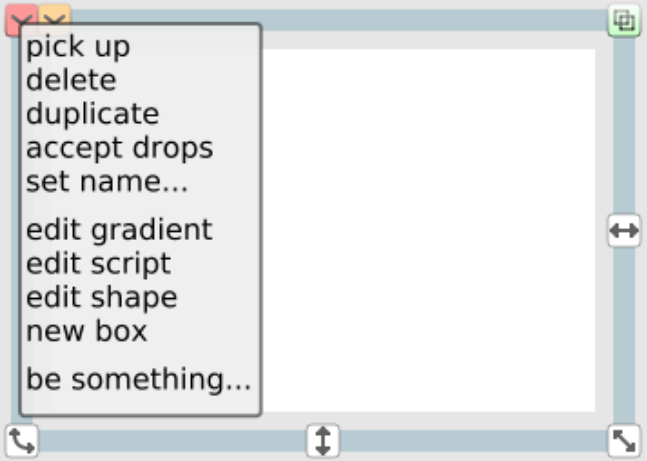
play stop

Don't cry over a dropped card deck!

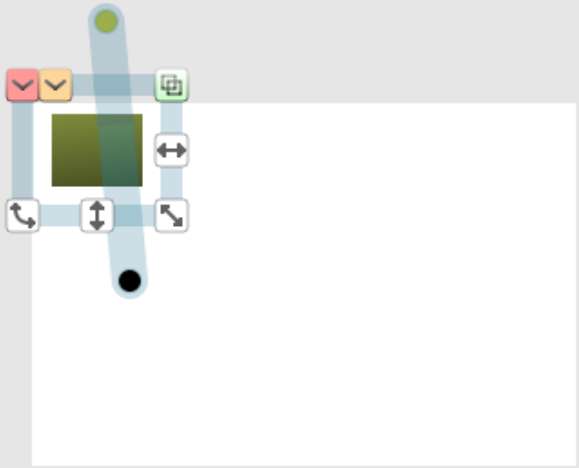
Ellis D. Kropotechev and Zeus,
A Marvelous Time-Sharing Device ('67)

play stop

Open the Oven now



Open the Oven now



Open the Oven now

be a button
be a slider
be a list
be a line of text
be a text field
be movable

Open the Oven now

back

new page copy page

delete page paste page

pages

KSBox(, 127@152)

+/-

Open the Oven now

KSBox(, 127@152)

+/-

fire <- anyE(contents, "fire")

fire

fire = undefined

fire <- anyE(contents, "fire")

Open the Oven now

Frank Document Editor

(Exit fullscreen mode.)

start

Document | In & Out | Content | Script

AGERE2013

back back

new page | copy page | delete page | paste page

pages

new | edit

threads

blank background | page like... | new document

blank page

Palatino Linotype | 10

default font

portrait/landscape | fullscreen | margin narrower | screen DPI | margin wider | page size | extra window

frame

- Inbox
- Sent
- Trash
- Draft
- Spam
- local
- remote
- ODF
- cache

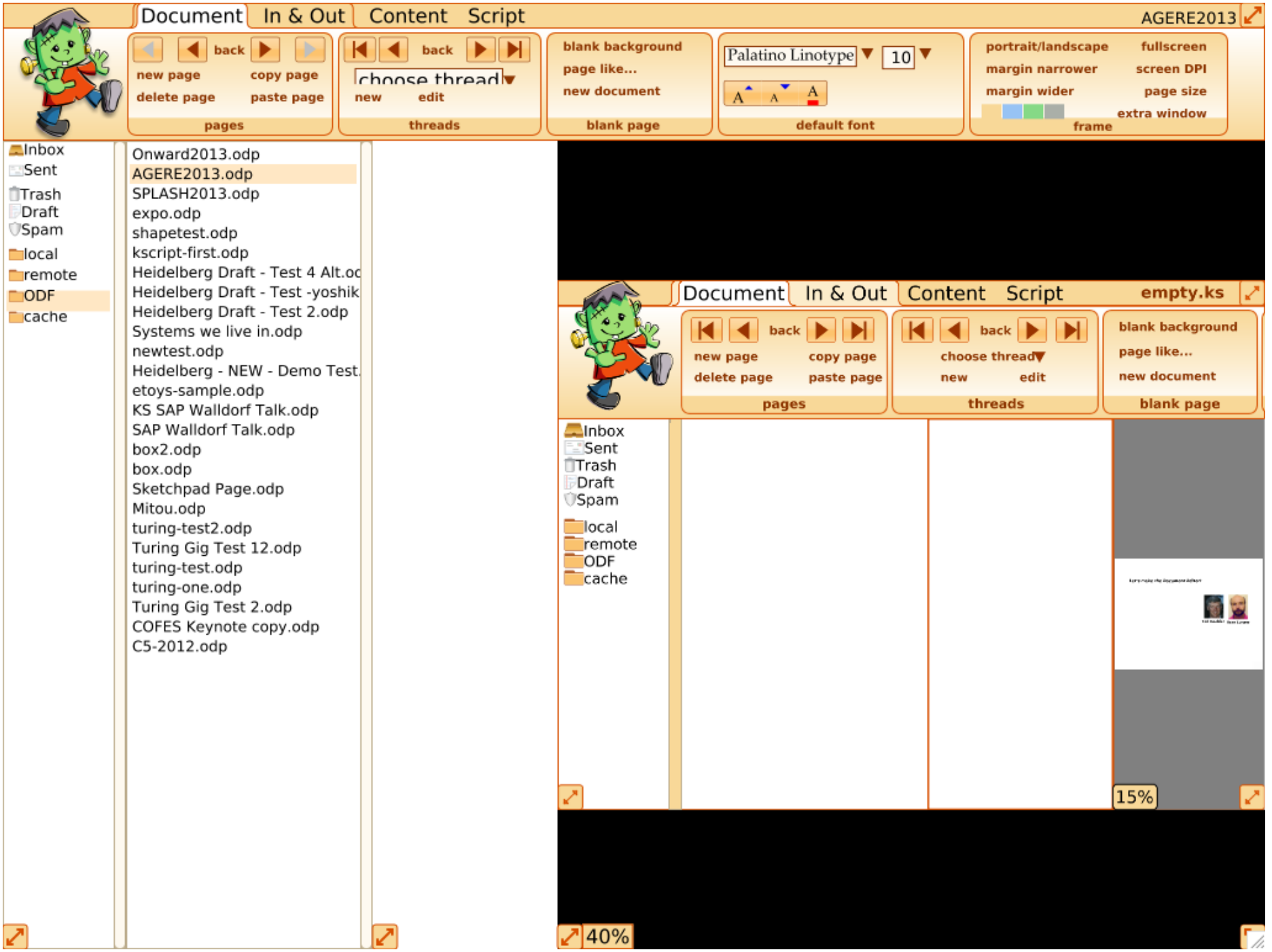
Onward2013.odp
AGERE2013.odp
SPLASH2013.odp
expo.odp
shapetest.odp
kscript-first.odp
Heidelberg Draft - Test 4 Alt.oc
Heidelberg Draft - Test -yoshik
Heidelberg Draft - Test 2.odp
Systems we live in.odp
newtest.odp
Heidelberg - NEW - Demo Test.
etoys-sample.odp
KS SAP Walldorf Talk.odp
SAP Walldorf Talk.odp
box2.odp
box.odp
Sketchpad Page.odp
Mitou.odp
turing-test2.odp
Turing Gig Test 12.odp
turing-test.odp
turing-one.odp
Turing Gig Test 2.odp
COFES Keynote copy.odp
C5-2012.odp

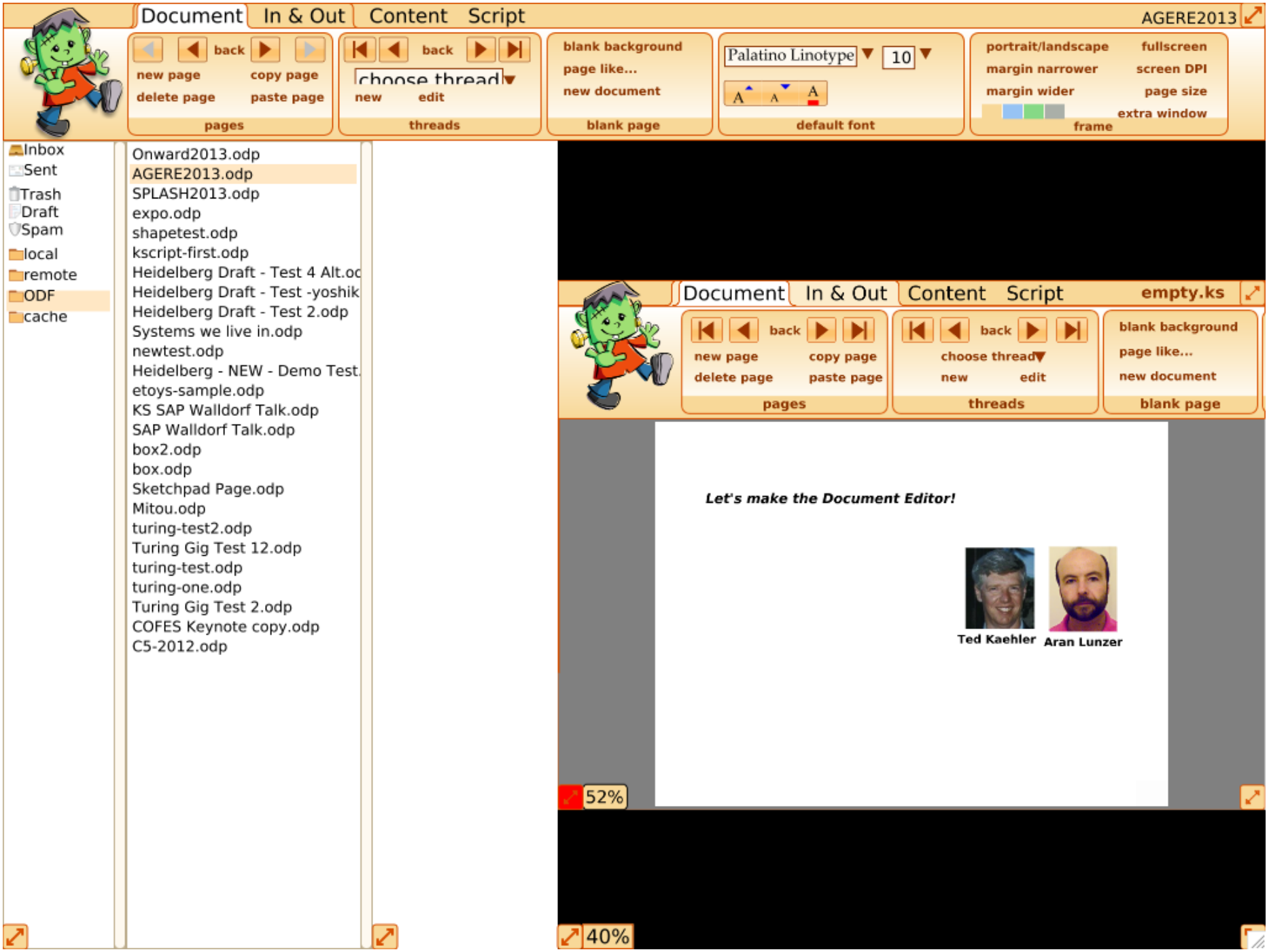
Frank Document Editor

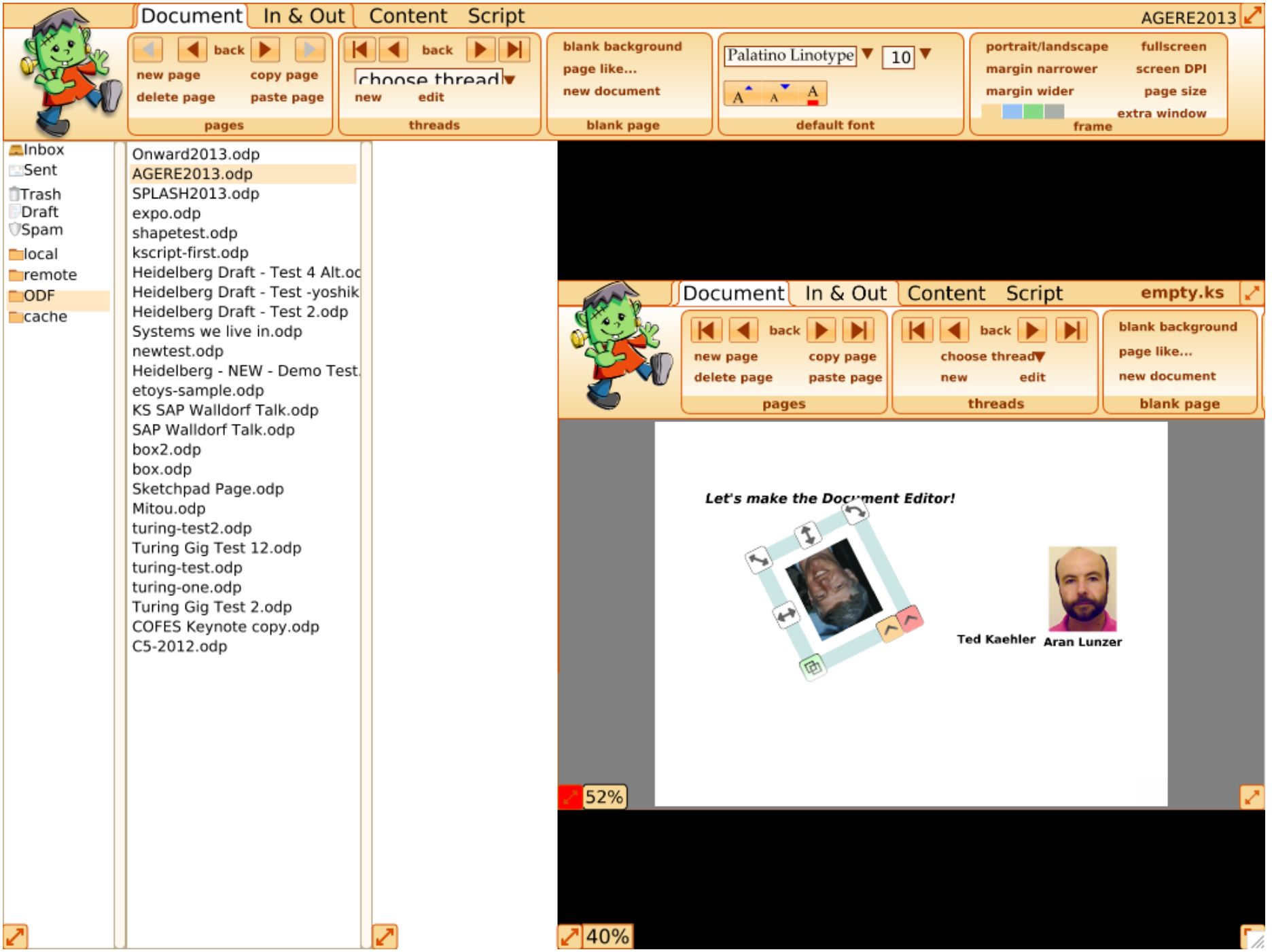
(Exit fullscreen mode.)

start

40%


Document
In & Out
Content
Script
AGERE2013
back
new page
copy page
delete page
paste page
pages
back
choose thread
new
edit
threads
blank background
page like...
new document
blank page
Palatino Linotype
10
default font
portrait/landscape
fullscreen
margin narrower
screen DPI
margin wider
page size
extra window
frame
Inbox
Sent
Trash
Draft
Spam
local
remote
ODF
cache
Onward2013.odp
AGERE2013.odp
SPLASH2013.odp
expo.odp
shapetest.odp
kscript-first.odp
Heidelberg Draft - Test 4 Alt.oc
Heidelberg Draft - Test -yoshik
Heidelberg Draft - Test 2.odp
Systems we live in.odp
newtest.odp
Heidelberg - NEW - Demo Test.
etoys-sample.odp
KS SAP Walldorf Talk.odp
SAP Walldorf Talk.odp
box2.odp
box.odp
Sketchpad Page.odp
Mitou.odp
turing-test2.odp
Turing Gig Test 12.odp
turing-test.odp
turing-one.odp
Turing Gig Test 2.odp
COFES Keynote copy.odp
C5-2012.odp
Document
In & Out
Content
Script
empty.ks
back
new page
copy page
delete page
paste page
pages
back
choose thread
new
edit
threads
blank background
page like...
new document
blank page
Inbox
Sent
Trash
Draft
Spam
local
remote
ODF
cache
15%
40%


Document
In & Out
Content
Script
AGERE2013
back
new page
copy page
delete page
paste page
pages
back
choose thread
new
edit
threads
blank background
page like...
new document
blank page
Palatino Linotype
10
default font
portrait/landscape
fullscreen
margin narrower
screen DPI
margin wider
page size
extra window
frame
Inbox
Sent
Trash
Draft
Spam
local
remote
ODF
cache
Onward2013.odp
AGERE2013.odp
SPLASH2013.odp
expo.odp
shapetest.odp
kscript-first.odp
Heidelberg Draft - Test 4 Alt.oc
Heidelberg Draft - Test -yoshik
Heidelberg Draft - Test 2.odp
Systems we live in.odp
newtest.odp
Heidelberg - NEW - Demo Test.
etoys-sample.odp
KS SAP Walldorf Talk.odp
SAP Walldorf Talk.odp
box2.odp
box.odp
Sketchpad Page.odp
Mitou.odp
turing-test2.odp
Turing Gig Test 12.odp
turing-test.odp
turing-one.odp
Turing Gig Test 2.odp
COFES Keynote copy.odp
C5-2012.odp
Document
In & Out
Content
Script
empty.ks
back
new page
copy page
delete page
paste page
pages
back
choose thread
new
edit
threads
blank background
page like...
new document
blank page
Let's make the Document Editor!
Ted Kaehler
Aran Lunzer
52%
40%


Document
In & Out
Content
Script
AGERE2013
back
new page
copy page
delete page
paste page
pages
back
choose thread
new
edit
threads
blank background
page like...
new document
blank page
Palatino Linotype
10
default font
portrait/landscape
fullscreen
margin narrower
screen DPI
margin wider
page size
extra window
frame
Inbox
Sent
Trash
Draft
Spam
local
remote
ODF
cache
Onward2013.odp
AGERE2013.odp
SPLASH2013.odp
expo.odp
shapetest.odp
kscript-first.odp
Heidelberg Draft - Test 4 Alt.oc
Heidelberg Draft - Test -yoshik
Heidelberg Draft - Test 2.odp
Systems we live in.odp
newtest.odp
Heidelberg - NEW - Demo Test.
etoys-sample.odp
KS SAP Walldorf Talk.odp
SAP Walldorf Talk.odp
box2.odp
box.odp
Sketchpad Page.odp
Mitou.odp
turing-test2.odp
Turing Gig Test 12.odp
turing-test.odp
turing-one.odp
Turing Gig Test 2.odp
COFES Keynote copy.odp
C5-2012.odp
Document
In & Out
Content
Script
empty.ks
back
new page
copy page
delete page
paste page
pages
back
choose thread
new
edit
threads
blank background
page like...
new document
blank page
Let's make the Document Editor!
Ted Kaehler
Aran Lunzer
52%
40%

Document | In & Out | Content | Script

empty.ks

back | new page | copy page | delete page | paste page — pages

back | choose thread | new | edit — threads

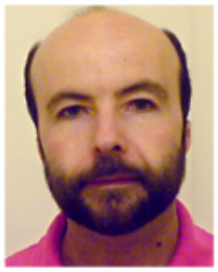
blank background | page like... | new document — blank page

PalatinoLinotype | 10 — default font

portrait/landscape | fullscreen | margin narrower | screen DPI | margin wider | page size | extra window — frame

find | abc | in: everyth | load

Let's make the Document Editor!



Ted Kaehler **Aran Lunzer**

109%

Document | In & Out | Content | Script

empty.ks

back | new page | copy page | delete page | paste page — pages

back | choose thread | new | edit — threads

blank background | page like... | new document — blank page

PalatinoLinotype | 10 — default font

portrait/landscape | fullscreen | margin narrower | screen DPI | margin wider | page size | extra window — frame

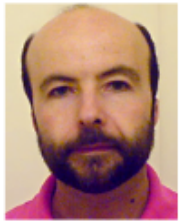
find | abc | in: everyth | load

Inbox
Sent
Trash
Draft
Spam

local
remote
ODF
cache

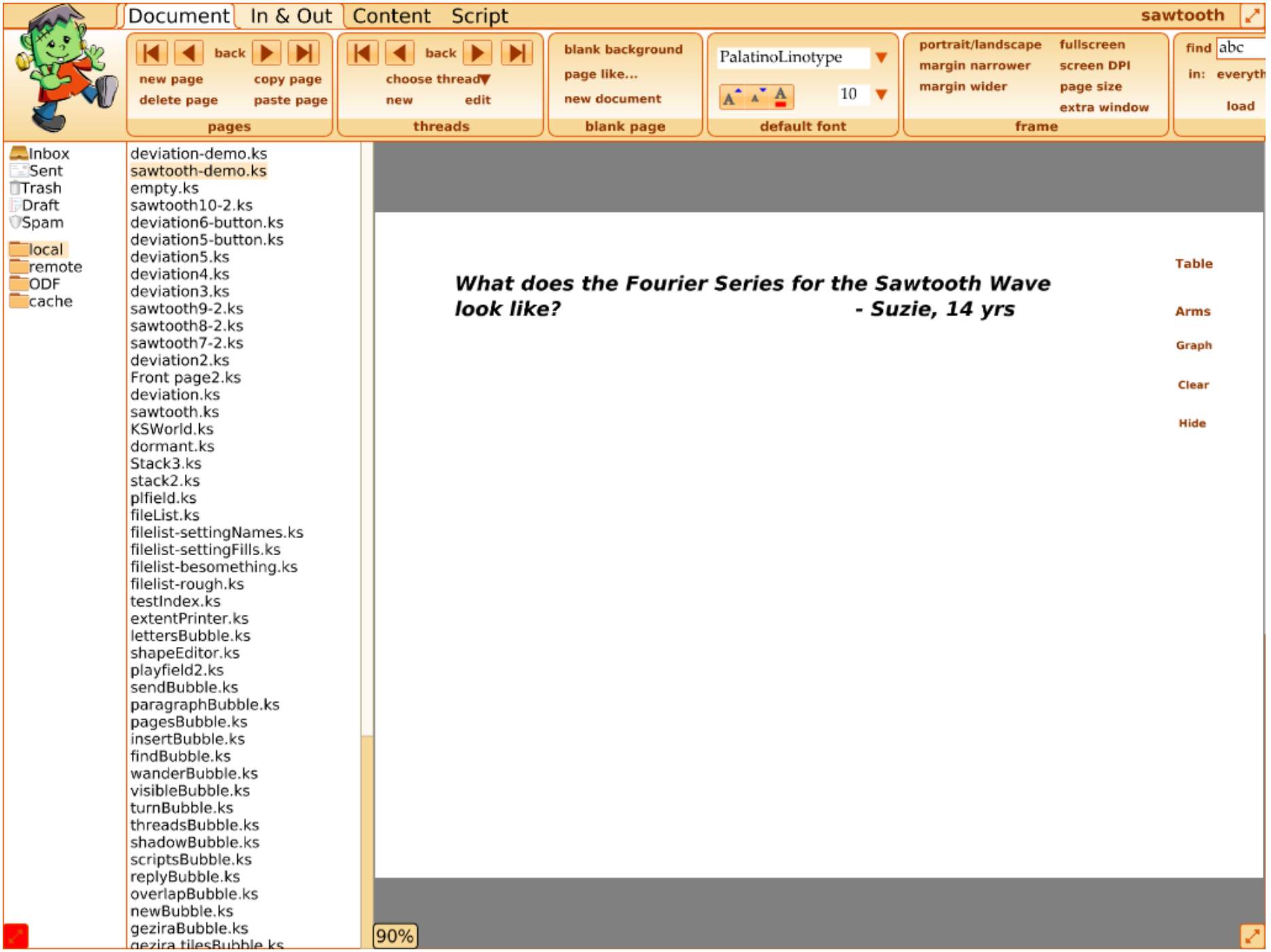
deviation-demo.ks
sawtooth-demo.ks
empty.ks
sawtooth10-2.ks
deviation6-button.ks
deviation5-button.ks
deviation5.ks
deviation4.ks
deviation3.ks
sawtooth9-2.ks
sawtooth8-2.ks
sawtooth7-2.ks
deviation2.ks
Front page2.ks
deviation.ks
sawtooth.ks
KSWorld.ks
dormant.ks
Stack3.ks
stack2.ks
plfield.ks
fileList.ks
filelist-settingNames.ks
filelist-settingFills.ks
filelist-besomething.ks
filelist-rough.ks
testIndex.ks
extentPrinter.ks
lettersBubble.ks
shapeEditor.ks
playfield2.ks
sendBubble.ks
paragraphBubble.ks
pagesBubble.ks
insertBubble.ks
findBubble.ks
wanderBubble.ks
visibleBubble.ks
turnBubble.ks
threadsBubble.ks
shadowBubble.ks
scriptsBubble.ks
replyBubble.ks
overlapBubble.ks
newBubble.ks
geziraBubble.ks
gezira tilesBubble.ks

Let's make the Document Editor!



Ted Kaehler **Aran Lunzer**

90%


Document
In & Out
Content
Script
sawtooth
back
new page
copy page
delete page
paste page
pages
back
choose thread
new
edit
threads
blank background
page like...
new document
blank page
PalatinoLinotype
10
default font
portrait/landscape
fullscreen
margin narrower
screen DPI
margin wider
page size
extra window
frame
find
abc
in:
everyth
load
Inbox
Sent
Trash
Draft
Spam
local
remote
ODF
cache
deviation-demo.ks
sawtooth-demo.ks
empty.ks
sawtooth10-2.ks
deviation6-button.ks
deviation5-button.ks
deviation5.ks
deviation4.ks
deviation3.ks
sawtooth9-2.ks
sawtooth8-2.ks
sawtooth7-2.ks
deviation2.ks
Front page2.ks
deviation.ks
sawtooth.ks
KSWorld.ks
dormant.ks
Stack3.ks
stack2.ks
plfield.ks
fileList.ks
filelist-settingNames.ks
filelist-settingFills.ks
filelist-besomething.ks
filelist-rough.ks
testIndex.ks
extentPrinter.ks
lettersBubble.ks
shapeEditor.ks
playfield2.ks
sendBubble.ks
paragraphBubble.ks
pagesBubble.ks
insertBubble.ks
findBubble.ks
wanderBubble.ks
visibleBubble.ks
turnBubble.ks
threadsBubble.ks
shadowBubble.ks
scriptsBubble.ks
replyBubble.ks
overlapBubble.ks
newBubble.ks
geziraBubble.ks
gezira tilesBubble.ks
What does the Fourier Series for the Sawtooth Wave look like? - Suzie, 14 yrs
Table
Arms
Graph
Clear
Hide
90%

What does the Fourier Series for the Sawtooth Wave look like? ***- Suzie, 14 yrs***

Table

Arms

Graph

Clear

Hide

Sawtooth

	heading	length	speed	timer
1	0.00	100.00	0.03	0.00
2	0.00	50.00	-0.09	
3	0.00	33.33	0.15	
4	0.00	25.00	-0.21	
5	0.00	20.00	0.27	

<-

vtooth Wave
ie, 14 yrs

Table

Arms

Graph

Clear

Hide

Sawtooth

	heading	length	speed	timer
1	0.00	100.00	0.03	0.00
2	0.00	50.00	-0.09	
3	0.00	33.33	0.15	
4	0.00	25.00	-0.21	
5	0.00	20.00	0.27	

heading2 <- **0.0 fby heading2' + speed2 on timer1**

vtooth Wave
ie, 14 yrs

Table

Arms

Graph

Clear

Hide

Sawtooth

	heading	length	speed	timer
1	0.00	100.00	0.03	0.00
2	0.00	50.00	-0.09	
3	0.00	33.33	0.15	
4	0.00	25.00	-0.21	
5	0.00	20.00	0.27	

heading2 <- **0.0 fby heading2' + speed2 on timer1**

vtooth Wave
ie, 14 yrs

Table

Arms

Graph

Clear

Hide

Sawtooth

		heading	length	speed	timer
1		0.00	100.00	0.03	0.00
2		0.00	50.00	-0.09	
3		0.00	33.33	0.15	
4		0.00	25.00	-0.21	
5		0.00	20.00	0.27	

timer1 <- **timerE(100)**

vtooth Wave
ie, 14 yrs

Table

Arms

Graph

Clear

Hide

Sawtooth

	heading	length	speed	timer
1	7.83	100.00	0.03	178600.00
2	-23.49	50.00	-0.09	
3	39.15	33.33	0.15	
4	-54.81	25.00	-0.21	
5	70.47	20.00	0.27	

timer1 <- **timerE(100)**

vtooth Wave
ie, 14 yrs

Table

Arms

Graph

Clear

Hide

Sawtooth

	heading	length	speed	timer
1	12.42	100.00	0.03	200800.00
2	31.85	50.00	-0.09	
3	62.10	33.33	0.15	
4	-86.94	25.00	-0.21	
5	111.78	20.00	0.27	

heading2 <- **0.0 fby heading2' + speed2 + 1 on timer1**

vtooth Wave
ie, 14 yrs

Table

Arms

Graph

Clear

Hide

Document | In & Out | Content | Script

deviation6-button.ks

back | new page | copy page | delete page | paste page — pages

back | choose thread | new | edit — threads

blank background | page like... | new document — blank page

PalatinoLinotype | 10 — default font

portrait/landscape | fullscreen | margin narrower | screen DPI | margin wider | page size | extra window — frame

find | abc | in: everyth | load

Inbox
Sent
Trash
Draft
Spam

local
remote
ODF
cache

deviation-demo.ks
sawtooth-demo.ks
empty.ks
sawtooth10-2.ks
deviation6-button.ks
deviation5-button.ks
deviation5.ks
deviation4.ks
deviation3.ks
sawtooth9-2.ks
sawtooth8-2.ks
sawtooth7-2.ks
deviation2.ks
Front page2.ks
deviation.ks
sawtooth.ks
KSWorld.ks
dormant.ks
Stack3.ks
stack2.ks
plfield.ks
fileList.ks
filelist-settingNames.ks
filelist-settingFills.ks
filelist-besomething.ks
filelist-rough.ks
testIndex.ks
extentPrinter.ks
lettersBubble.ks
shapeEditor.ks
playfield2.ks
sendBubble.ks
paragraphBubble.ks
pagesBubble.ks
insertBubble.ks
findBubble.ks
wanderBubble.ks
visibleBubble.ks
turnBubble.ks
threadsBubble.ks
shadowBubble.ks
scriptsBubble.ks
replyBubble.ks
overlapBubble.ks
newBubble.ks
geziraBubble.ks
gezira tilesBubble.ks

The Standard Deviation is a widely used measurement of variability or diversity of a set of measurements. Volitility in the stock market is the Standard Deviation of the value of a stock. Standard Deviation shows how much variation or "dispersion" there is from the "average" (the mean). A low standard deviation indicates that the data points tend to be very close to the mean, whereas high standard deviation indicates that the data is spread out over a large range of values.

A useful property of Standard Deviation is that, unlike the variance, it is expressed in the same units as the data.

Let's use the vertical position of the sliders below as the data values. Move the sliders up and down.

The first step is to find the average of the data samples. The average is the sum of the values divided by the number of values. The sum of the values is 2.0 + 10.0 + 0.0 + 6.0 + 16.0 + 6.0 + -1.0 which is 39.0. There are 7 data values. The average is 5.571.

Now that we know the average, we can find out how much each sample differs from the average. Subtract the average from each sample. We need the square of each difference. Add up the squares 12.75 + 19.61 + 31.04 + 0.18 + 108.7 + 0.18 + 43.18 and take the average of those to get 30.81. This called the variance. Taking the square root gives a Standard Deviation of 5.550. It is in the same units as the data values. For a normal distribution (or bell curve) of data, 68.2% of the values will be within one standard deviation of the average.

next | twinkle

105%

The Standard Deviation is a widely used measurement of variability or diversity of a set of measurements. Volitility in the stock market is the Standard Deviation of the value of a stock. Standard Deviation shows how much variation or "dispersion" there is from the "average" (the mean). A low standard deviation indicates that the data points tend to be very close to the mean, whereas high standard deviation indicates that the data is spread out over a large range of values.

A useful property of Standard Deviation is that, unlike the variance, it is expressed in the same units as the data.

Let's use the vertical position of the sliders below as the data values. Move the sliders up and down.

The first step is to find the average of the data samples. The average is the sum of the values divided by the number of values. The sum of the values is 2.0 + 10.0 + 0.0 + 6.0 + 16.0 + 6.0 + -1.0 which is 39.0. There are 7 data values. The average is 5.571.

Now that we know the average, we can find out how much each sample differs from the average. Subtract the average from each sample. We need the square of each difference. Add up the squares 12.75 + 19.61 + 31.04 + 0.18 + 108.7 + 0.18 + 43.18 and take the average of those to get 30.81. This called the variance. Taking the square root gives a Standard Deviation of 5.550. It is in the same units as the data values. For a normal distribution (or bell curve) of data, 68.2% of the values will be within one standard deviation of the average.

next twinkle

The Standard Deviation is a widely used measurement of variability or diversity of a set of measurements. Volitility in the stock market is the Standard Deviation of the value of a stock. Standard Deviation shows how much variation or "dispersion" there is from the "average" (the mean). A low standard deviation indicates that the data points tend to be very close to the mean, whereas high standard deviation indicates that the data is spread out over a large range of values.

A useful property of Standard Deviation is that, unlike the variance, it is expressed in the same units as the data.

Let's use the vertical position of the sliders below as the data values. Move the sliders up and down.

The first step is to find the average of the data samples. The average is the sum of the values divided by the number of values. The sum of the values is 2.0 + 10.0 + 0.0 + 6.0 + 16.0 + 6.0 + 14.0 which is 54.0. There are 7 data values. The average is 7.714.

Now that we know the average, we can find out how much each sample differs from the average. Subtract the average from each sample. We need the square of each difference. Add up the squares 32.65 + 5.22 + 59.51 + 2.93 + 68.65 + 2.93 + 39.51 and take the average of those to get 30.2. This called the variance. Taking the square root gives a Standard Deviation of 5.495. It is in the same units as the data values. For a normal distribution (or bell curve) of data, 68.2% of the values will be within one standard deviation of the average.

next twinkle

The Standard Deviation is a widely used measurement of variability or diversity of a set of measurements. Volitility in the stock market is the Standard Deviation of the value of a stock. Standard Deviation shows how much variation or "dispersion" there is from the "average" (the mean). A low standard deviation indicates that the data points tend to be very close to the mean, whereas high standard deviation indicates that the data is spread out over a large range of values.

A useful property of Standard Deviation is that, unlike the variance, it is expressed in the same units as the data.

Let's use the vertical position of the sliders below as the data values. Move the sliders up and down.

The first step is to find the average of the data samples. The average is the sum of the values divided by the number of values. The sum of the values is 2.0 + 10.0 + 0.0 + 6.0 + 16.0 + -6.0 + 14.0 which is 42.0. There are 7 data values. The average is 6.0.

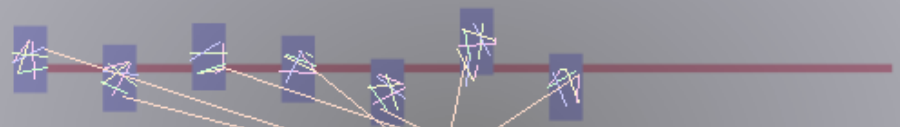
Now that we know the average, we can find out how much each sample differs from the average. Subtract the average from each sample. We need the square of each difference. Add up the squares 16.0 + 16.0 + 36.0 + 0.0 + 100.0 + 144.0 + 64.0 and take the average of those to get 53.71. This called the variance. Taking the square root gives a Standard Deviation of 7.329. It is in the same units as the data values. For a normal distribution (or bell curve) of data, 68.2% of the values will be within one standard deviation of the average.

next twinkle

The Standard Deviation is a widely used measurement of variability or diversity of a set of measurements. Volitility in the stock market is the Standard Deviation of the value of a stock. Standard Deviation shows how much variation or "dispersion" there is from the "average" (the mean). A low standard deviation indicates that the data points tend to be very close to the mean, whereas high standard deviation indicates that the data is spread out over a large range of values.

A useful property of Standard Deviation is that, unlike the variance, it is expressed in the same units as the data.

Let's use the vertical position of the sliders below as the data values. Move the sliders up and down.



The first step is to find the average of the data samples. The average is the sum of the values divided by the number of values. The sum of the values is 2.0 + 10.0 + 0.0 + 6.0 + 16.0 + 6.0 + 14.0 which is 42.0. There are 7 data values. The average is 6.0.

Now that we know the average, we can find out how much each sample differs from the average. Subtract the average from each sample. We need the square of each difference. Add up the squares 16.0 + 16.0 + 36.0 + 0.0 + 100.0 + 144.0 + 64.0 and take the average of those to get 53.71. This called the variance. Taking the square root gives a Standard Deviation of 7.329. It is in the same units as the data values. For a normal distribution (or bell curve) of data, 68.2% of the values will be within one standard deviation of the average.

next twinkle

Multiple Solvers

KScript is simple and powerful...

Multiple Solvers

KScript is simple and powerful...

But it's only one-way, equality based constraints.

- box layout needs multi-way constraints.

Multiple Solvers

KScript is simple and powerful...

But it's only one-way, equality based constraints.

- box layout needs multi-way constraints.

(Maybe we need different solvers and make them work together...)

Other Projects from STEPS

COLA: The combined object-lambda architecture.

Lively Kernel: An interactive system in browsers.

Tamacola: A meta-language environment in Flash.

Worlds: Controlling the scope of side effects.

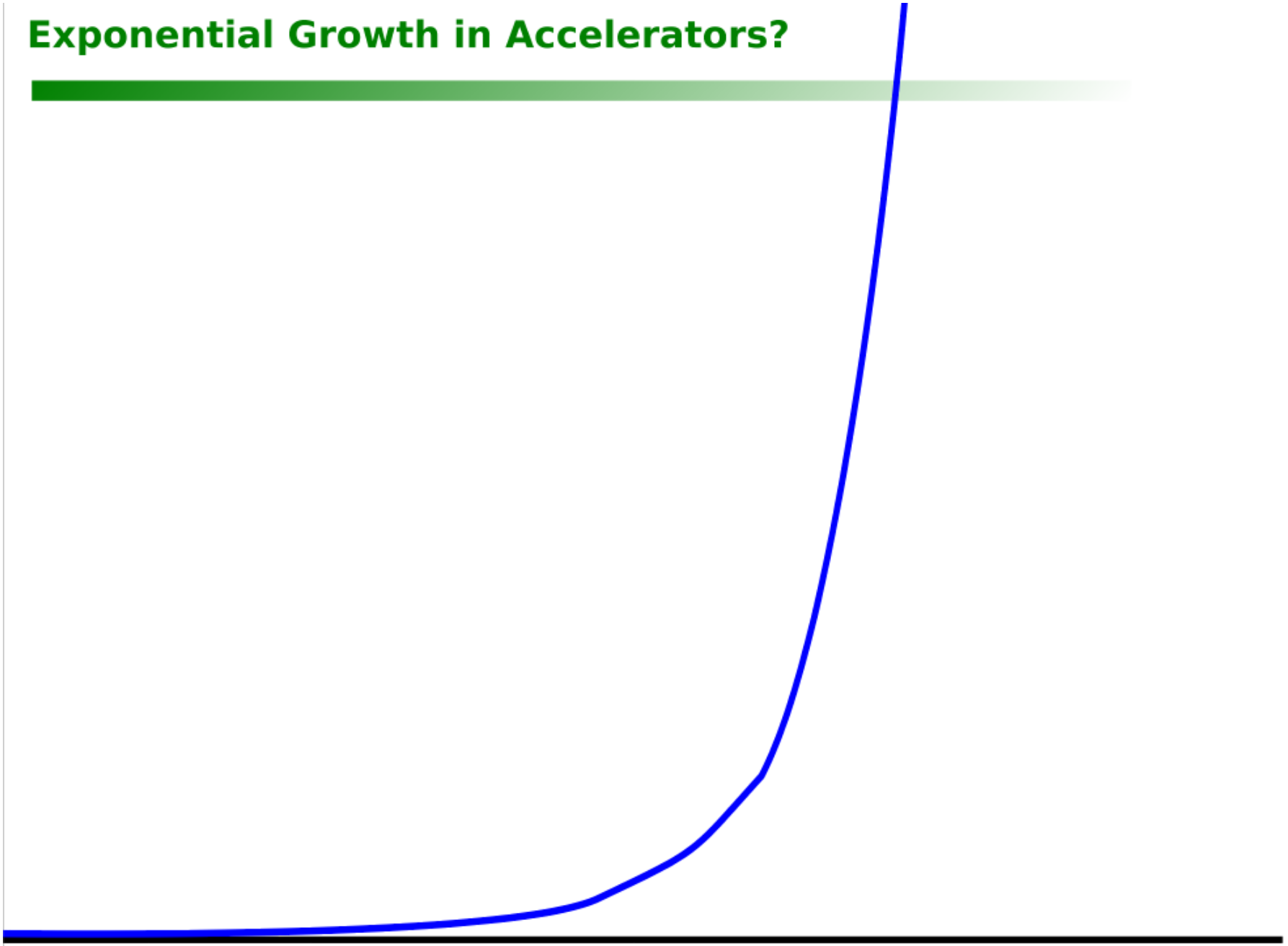
...

The Future?

The Future?

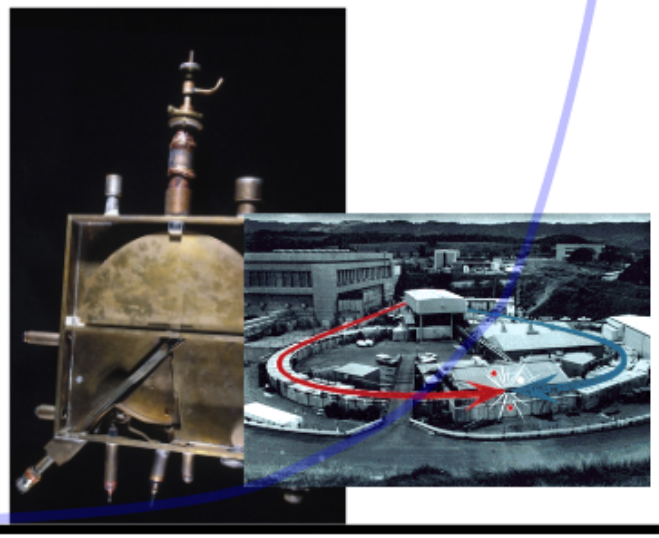
(Maybe there are the best way and also the easiest way to predict it?)

Exponential Growth in Accelerators?

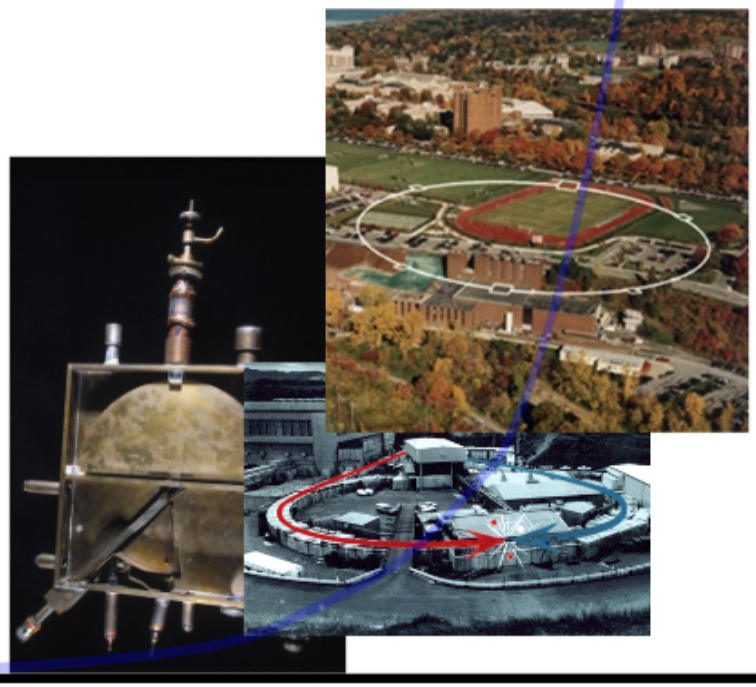


Exponential Growth in Accelerators?

Exponential Growth in Accelerators?



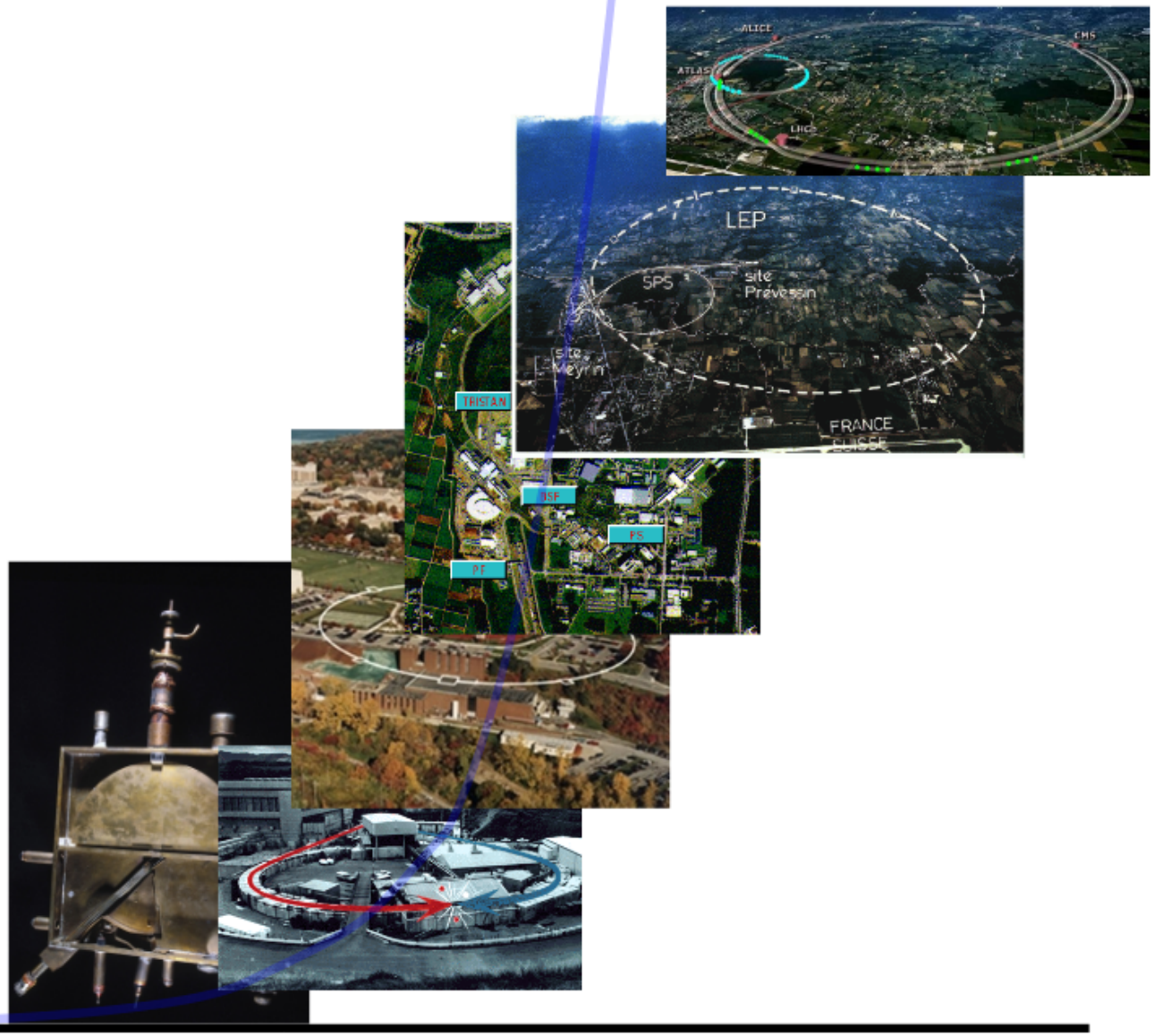
Exponential Growth in Accelerators?



Exponential Growth in Accelerators?

Exponential Growth in Accelerators?

Exponential Growth in Accelerators?



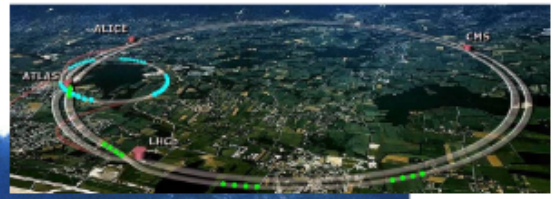
Exponential Growth in Accelerators?

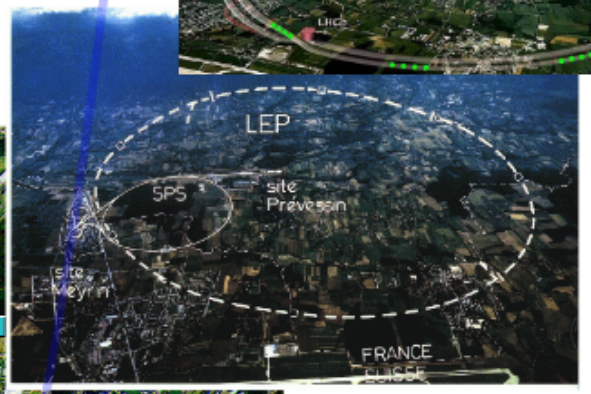
Exponential Growth in Accelerators?

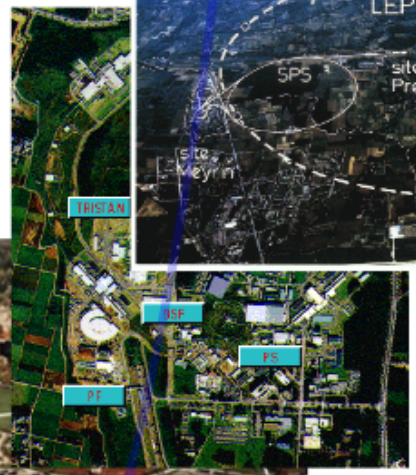
Leon Lederman

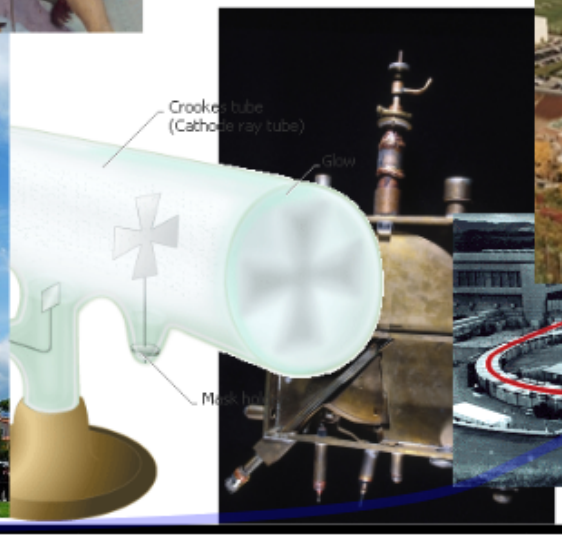
Exponential Growth in Accelerators?

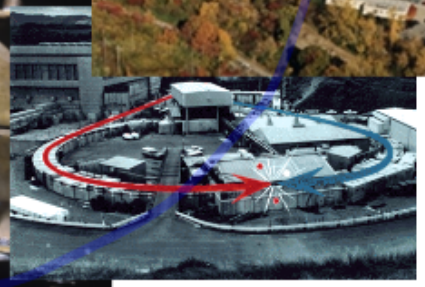
Leon Lederman





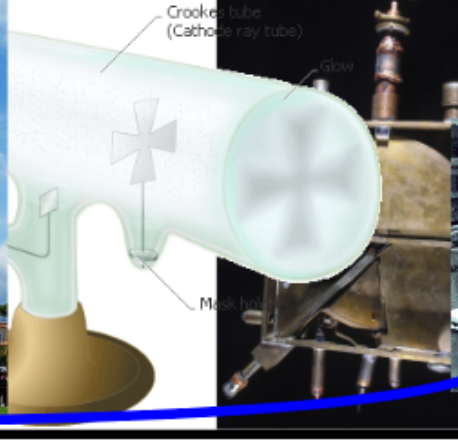


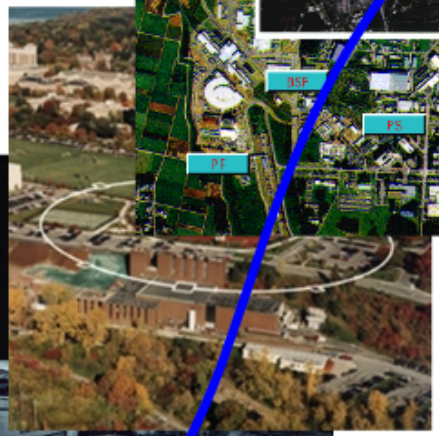


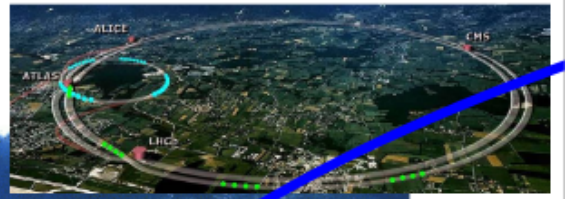


Exponential Growth in Accelerators?

Leon Lederman

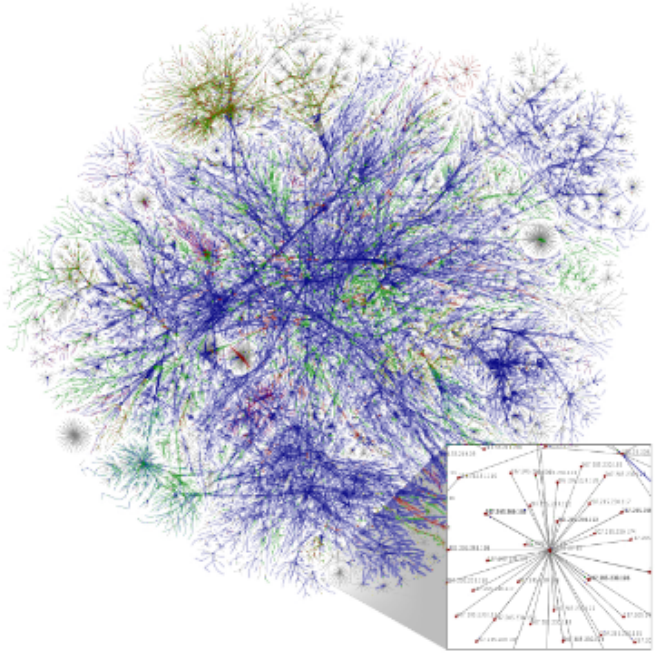






Exponential Growth of the Internet?

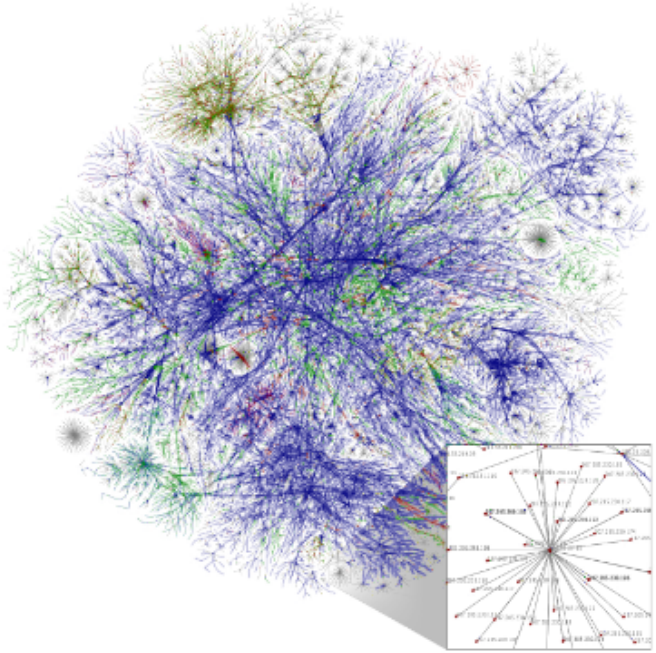
1984: The total computing capacity in the world < 1000 Haswell+GPU laptops



Exponential Growth of the Internet?

1984: The total computing capacity in the world < 1000 Haswell+GPU laptops

1984: The total data on the Internet < 2 TB

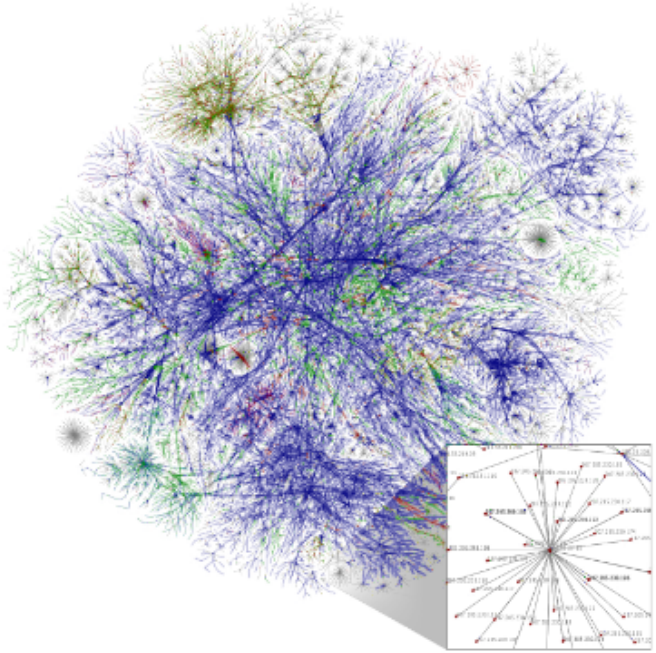


Exponential Growth of the Internet?

1984: The total computing capacity in the world < 1000 Haswell+GPU laptops

1984: The total data on the Internet < 2 TB

1996: The # of nodes on the Internet < 10 M

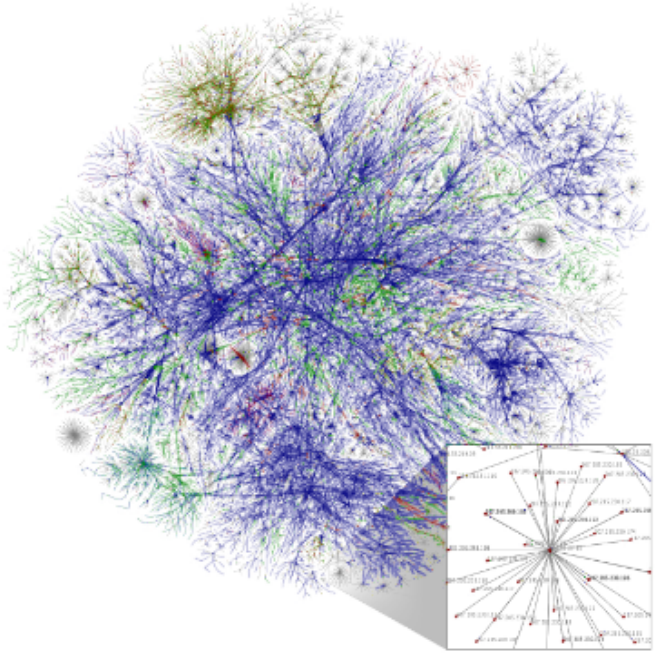


Exponential Growth of the Internet?

1984: The total computing capacity in the world < 1000 Haswell+GPU laptops

1984: The total data on the Internet < 2 TB

1996: The # of nodes on the Internet < 10 M



We can have a simulation of the Internet on a laptop!

The Internet all the way down!


Hesam Samimi


Alan Borning


Alex Warth


Todd Millstein

- "Share-nothing"

- No pointers

The Internet all the way down!


Hesam Samimi


Alan Borning


Alex Warth


Todd Millstein

- "Share-nothing"

- No pointers

- No “sending”

The Internet all the way down!


Hesam Samimi


Alan Borning


Alex Warth


Todd Millstein

- "Share-nothing"

- No pointers

- No “sending”

- Protection

The Internet all the way down!


Hesam Samimi


Alan Borning


Alex Warth


Todd Millstein

- "Share-nothing"

- No pointers

- No “sending”

- Protection

- Semantic discovery

Talking to totally uncorrelated “sapient” beings!

… Let us suppose that I am working at SDC, and that I find a program [written in FORTRAN] that looks suitable on a disc file in Berkeley. My programs were written in JOVIAL.

(MEMORANDUM FOR: Members and Affiliates of the Intergalactic Computer Network, Apr. 1963)

J.C.R. Licklider

Talking to totally uncorrelated “sapient” beings!

… Let us suppose that I am working at SDC, and that I find a program [written in FORTRAN] that looks suitable on a disc file in Berkeley. My programs were written in JOVIAL.

(MEMORANDUM FOR: Members and Affiliates of the Intergalactic Computer Network, Apr. 1963)

J.C.R. Licklider

Programs need to discover and find a way to talk to each other. (Agents?)

Talking to totally uncorrelated “sapient” beings!

… Let us suppose that I am working at SDC, and that I find a program [written in FORTRAN] that looks suitable on a disc file in Berkeley. My programs were written in JOVIAL.

(MEMORANDUM FOR: Members and Affiliates of the Intergalactic Computer Network, Apr. 1963)

J.C.R. Licklider

Programs need to discover and find a way to talk to each other. (Agents?)

Cyc

Very Super High Level Languages?

Very Super High Level Languages?

Very Super High Level Languages?

Discovery Engine

Very Super High Level Languages?

SMT solver

Discovery Engine

Very Super High Level Languages?

Constraints

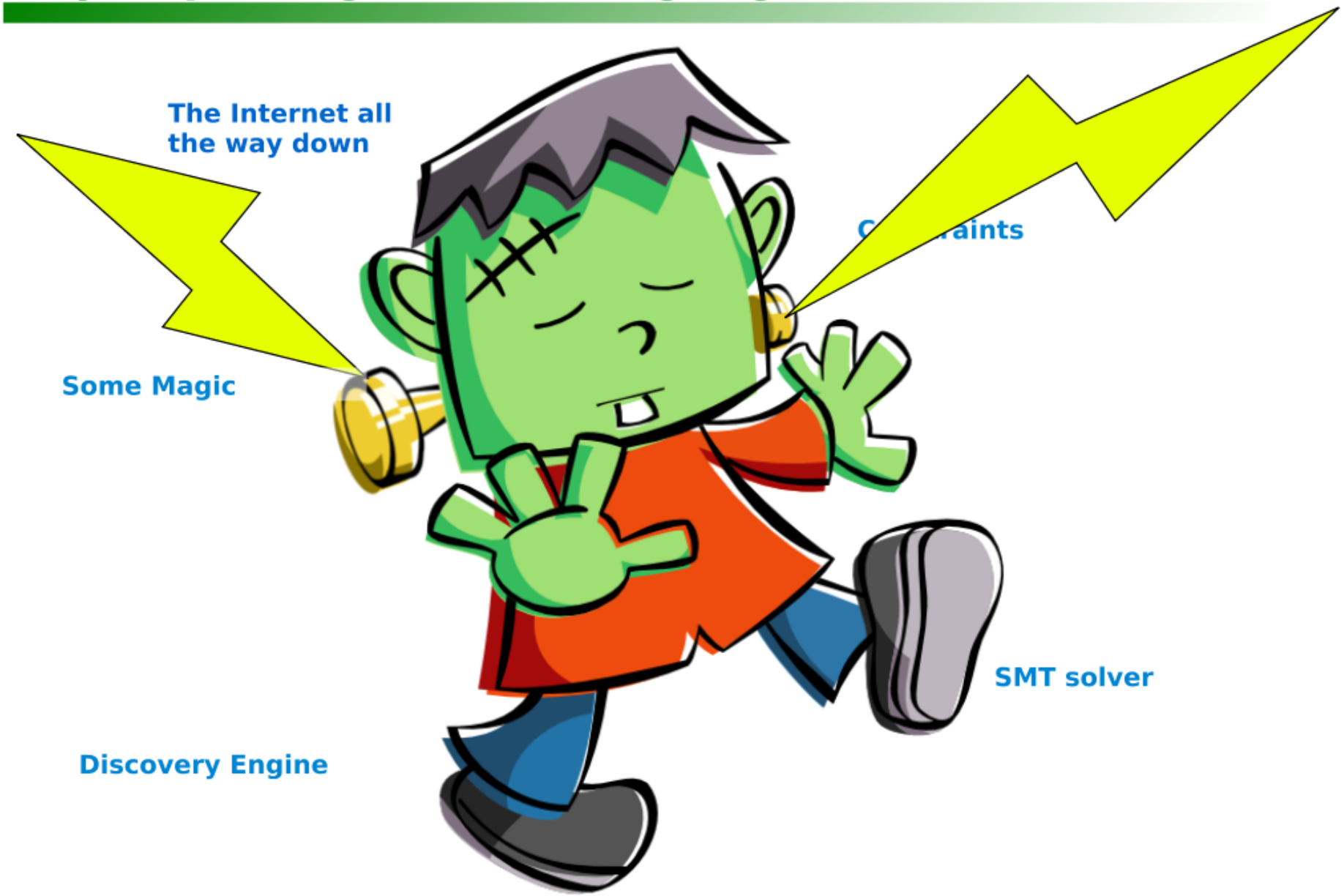
SMT solver

Discovery Engine

Very Super High Level Languages?

Very Super High Level Languages?

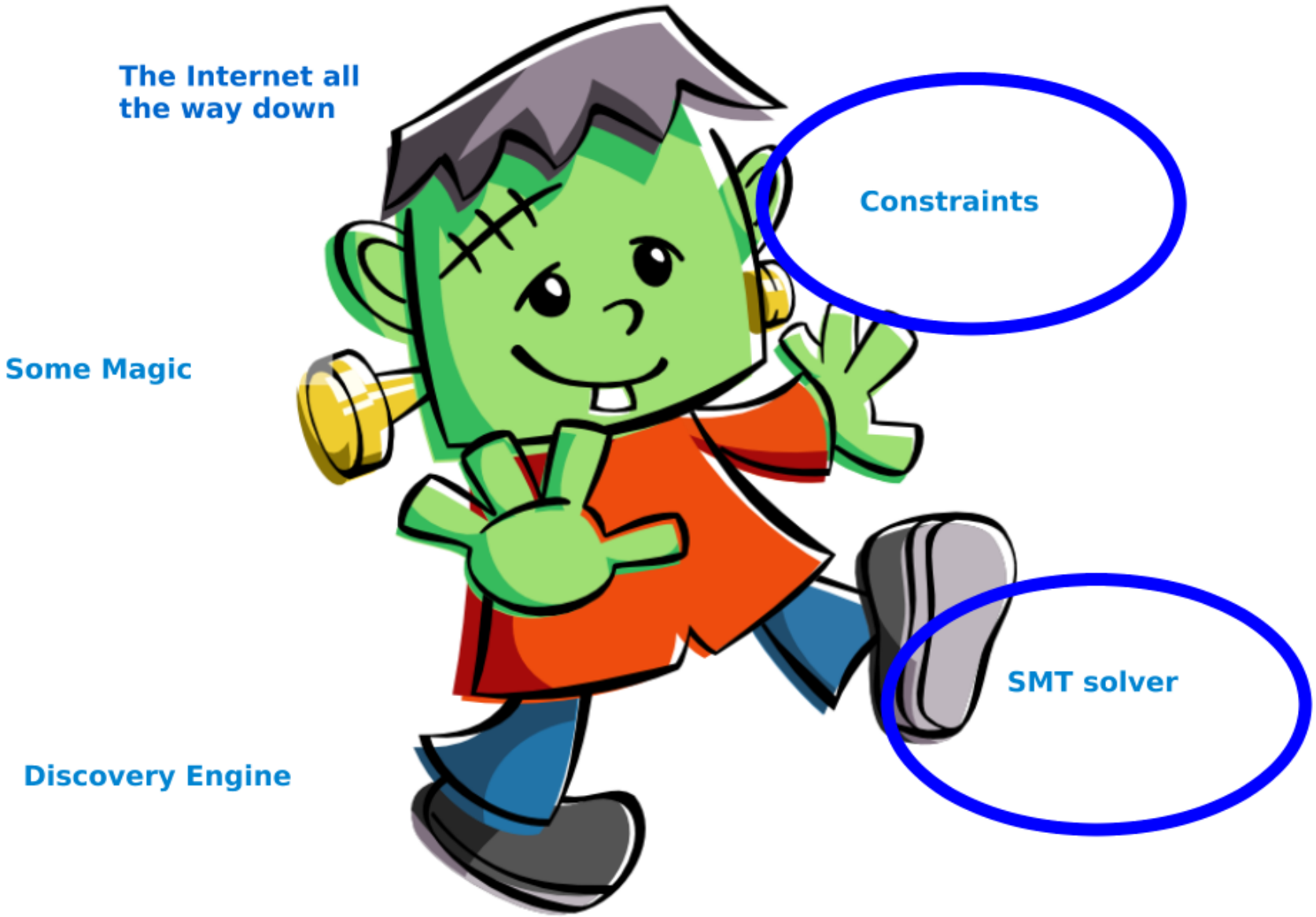
Very Super High Level Languages?



Very Super High Level Languages?

Very Super High Level Languages?

Very Super High Level Languages?



Architecture to go from "Whats" to "Hows"


Hesam Samimi
Alan Borning

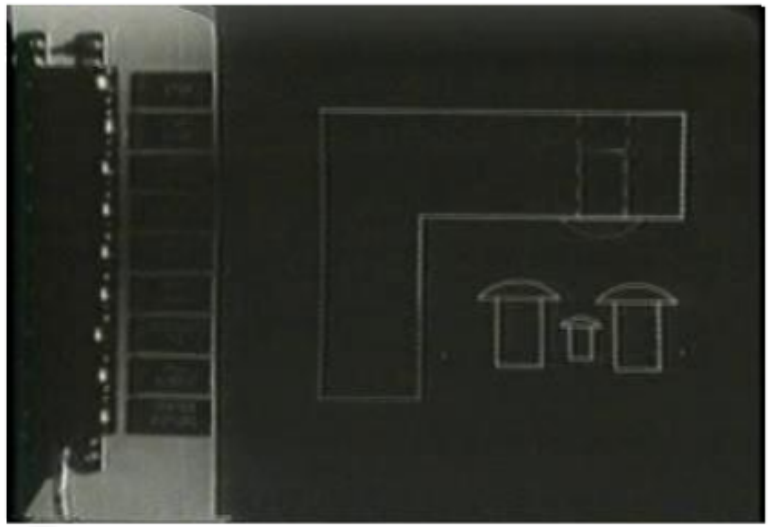
Solvers!

51st Birthday Of Sketchpad!

SKETCHPAD, A MAN-MACHINE GRAPHICAL COMMUNICATION SYSTEM

by

IVAN EDWARD SUTHERLAND



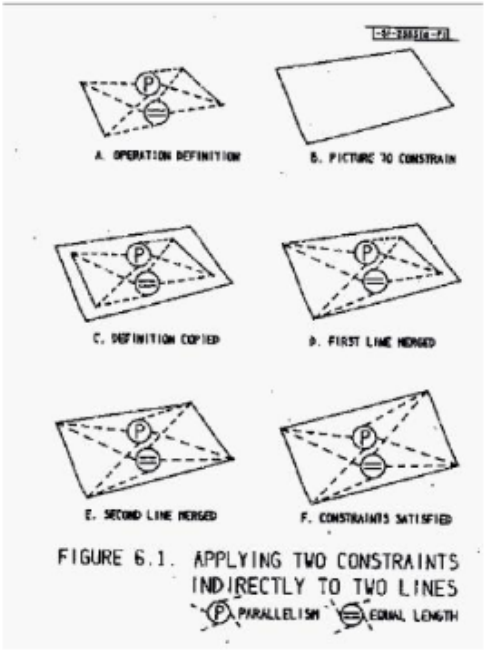
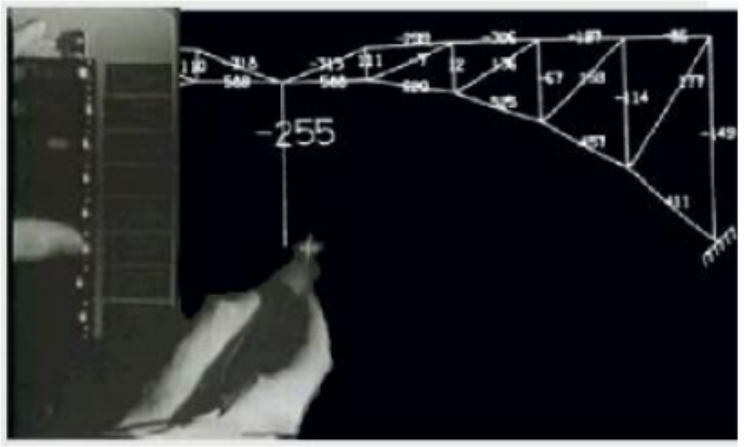


FIGURE 6.1. APPLYING TWO CONSTRAINTS INDIRECTLY TO TWO LINES

P PARALLELISM = EQUAL LENGTH

play

stop

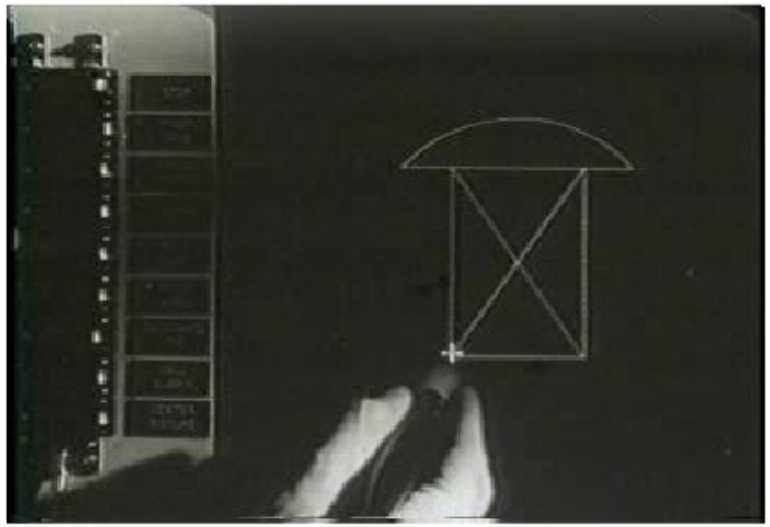


51st Birthday Of Sketchpad!

SKETCHPAD, A MAN-MACHINE GRAPHICAL COMMUNICATION SYSTEM

by

IVAN EDWARD SUTHERLAND



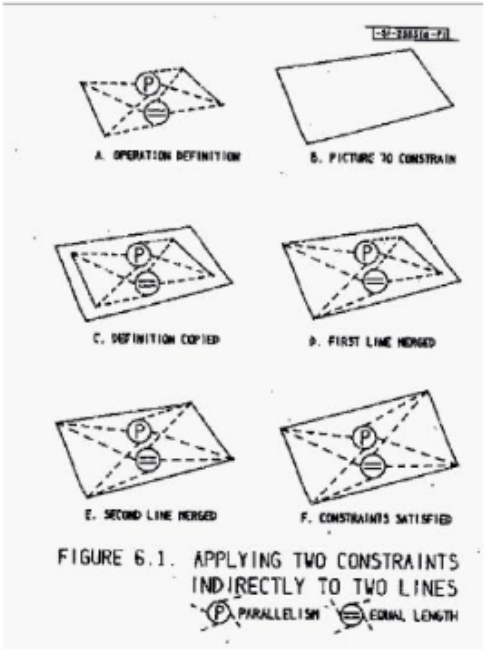
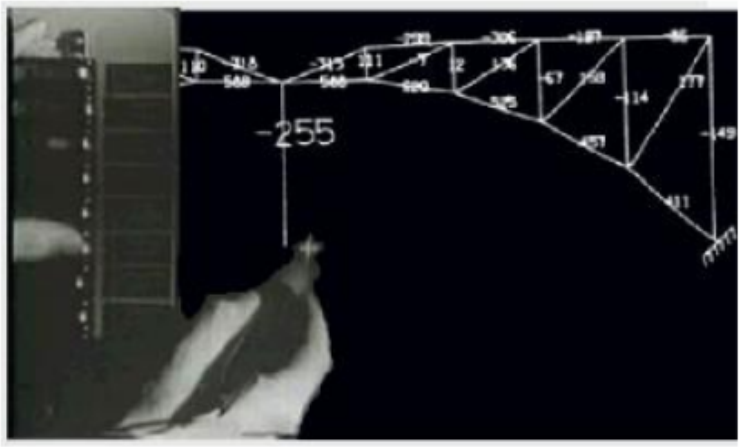


FIGURE 6.1. APPLYING TWO CONSTRAINTS INDIRECTLY TO TWO LINES

P PARALLELISM = EQUAL LENGTH

play

stop

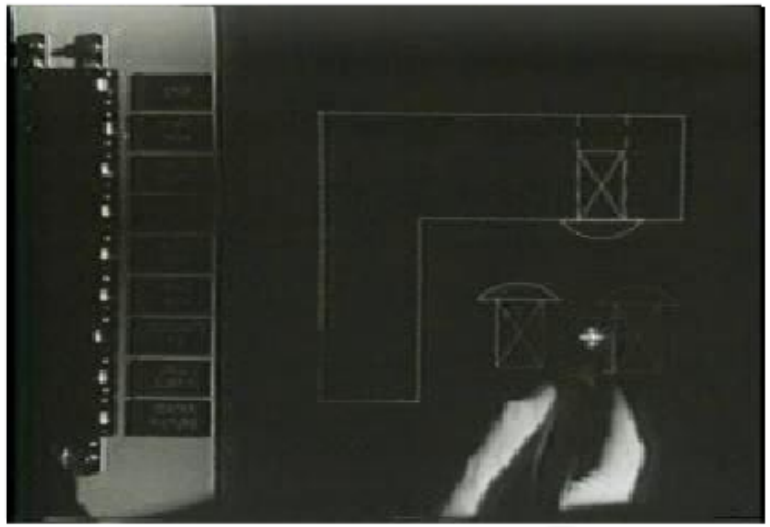


51st Birthday Of Sketchpad!

SKETCHPAD, A MAN-MACHINE GRAPHICAL COMMUNICATION SYSTEM

by

IVAN EDWARD SUTHERLAND



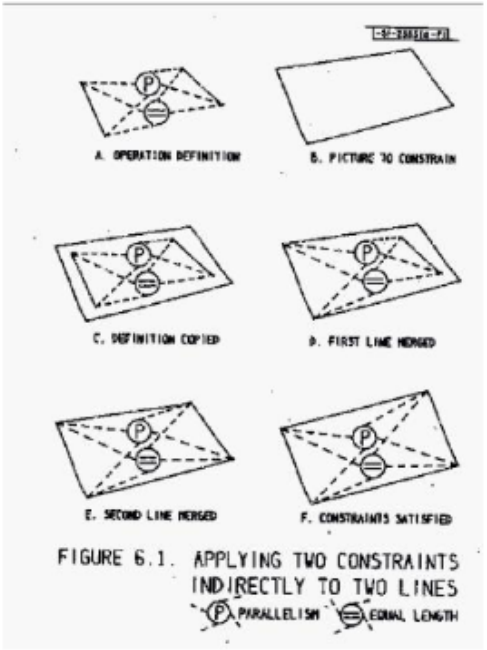
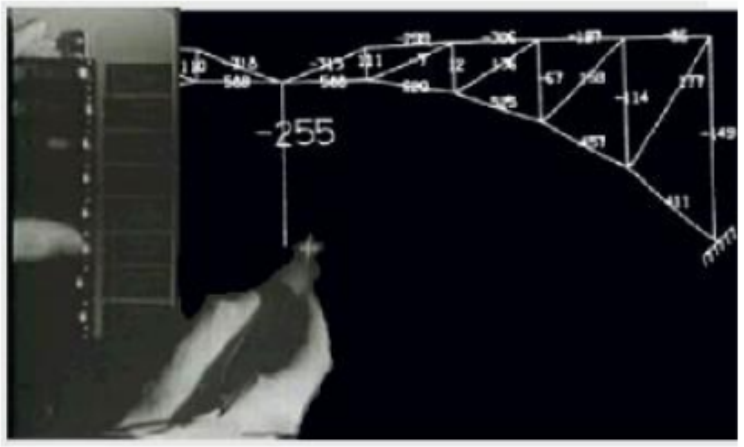


FIGURE 6.1. APPLYING TWO CONSTRAINTS INDIRECTLY TO TWO LINES

P PARALLELISM = EQUAL LENGTH

play

stop

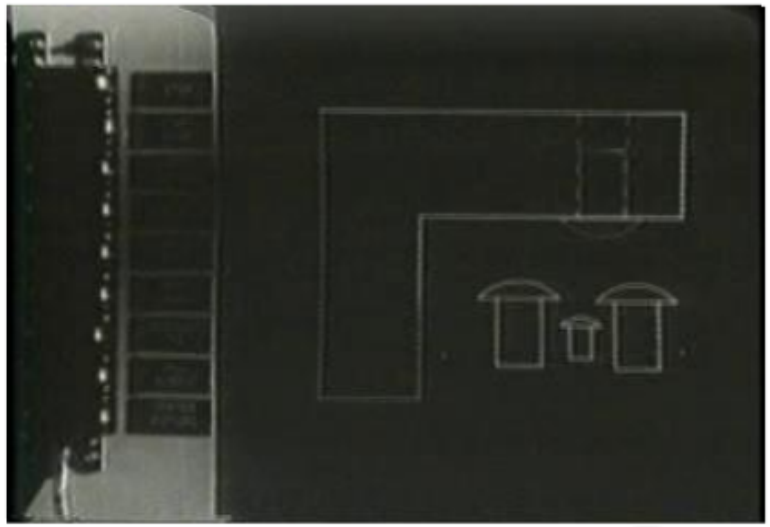


51st Birthday Of Sketchpad!

SKETCHPAD, A MAN-MACHINE GRAPHICAL COMMUNICATION SYSTEM

by

IVAN EDWARD SUTHERLAND



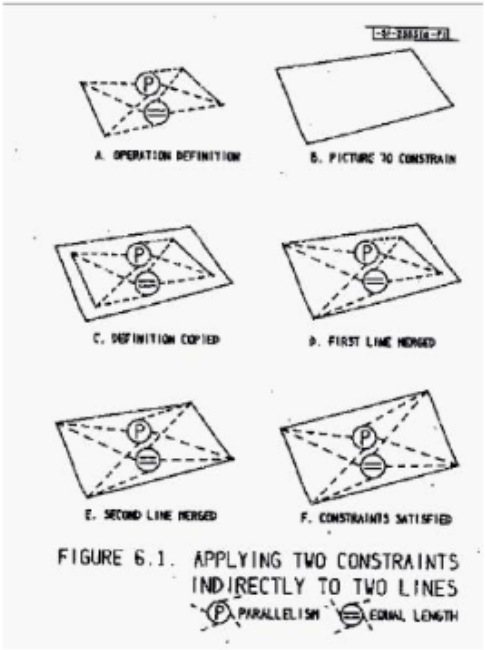
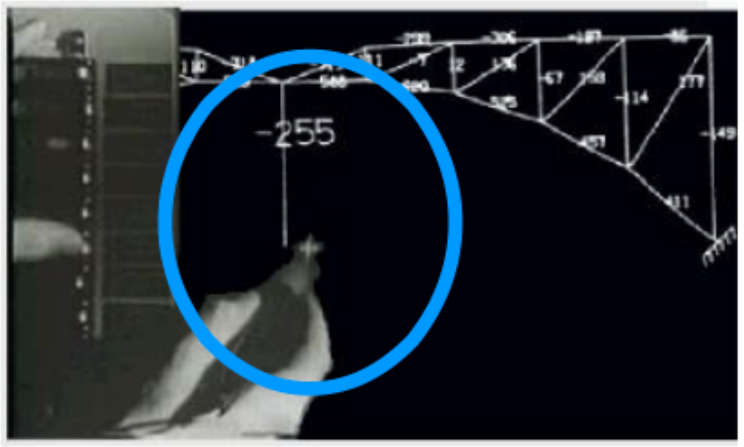


FIGURE 6.1. APPLYING TWO CONSTRAINTS INDIRECTLY TO TWO LINES

P PARALLELISM = EQUAL LENGTH

play

stop

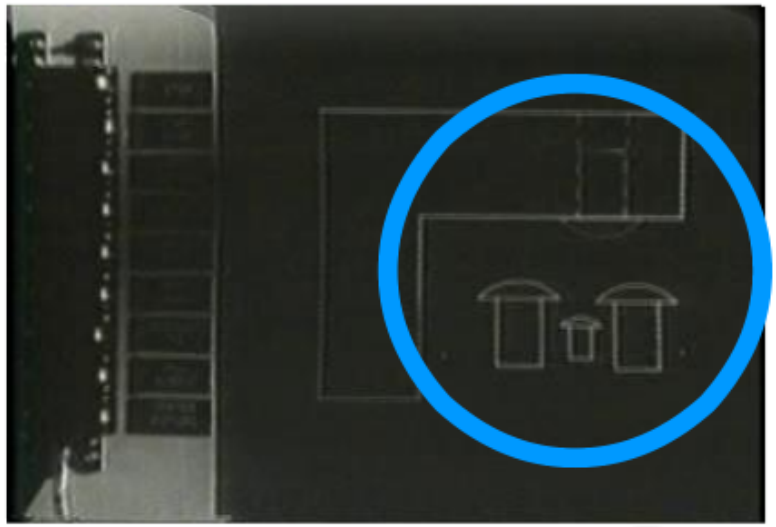


51st Birthday Of Sketchpad!

SKETCHPAD, A MAN-MACHINE GRAPHICAL COMMUNICATION SYSTEM

by

IVAN EDWARD SUTHERLAND



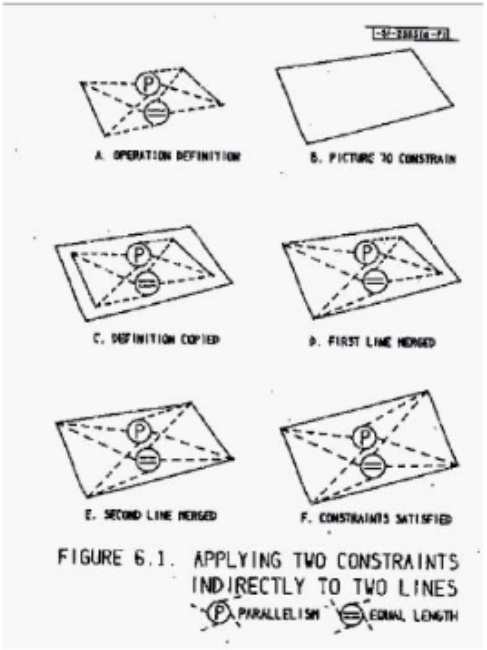
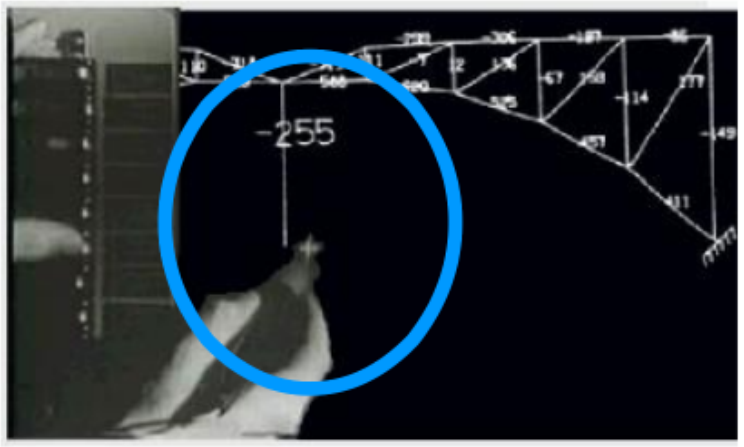


FIGURE 6.1. APPLYING TWO CONSTRAINTS INDIRECTLY TO TWO LINES

P PARALLELISM = EQUAL LENGTH

play

stop

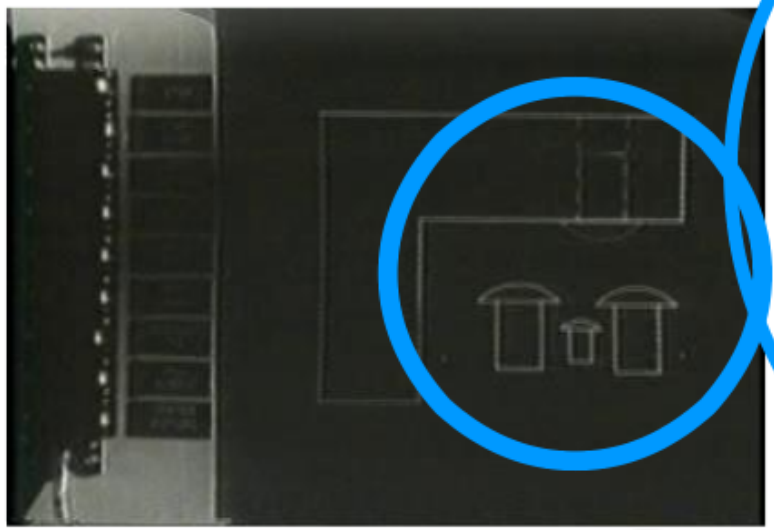


51st Birthday Of Sketchpad!

SKETCHPAD, A MAN-MACHINE GRAPHICAL COMMUNICATION SYSTEM

by

IVAN EDWARD SUTHERLAND



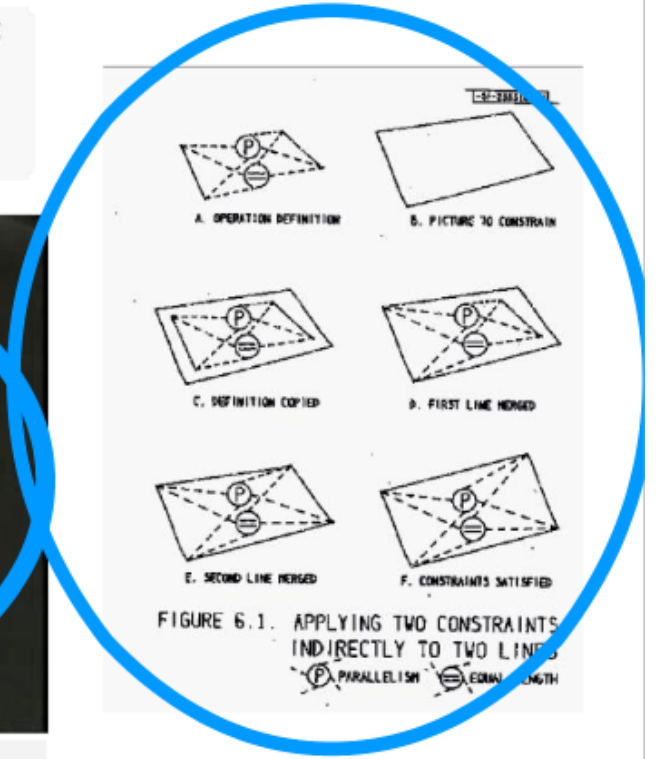
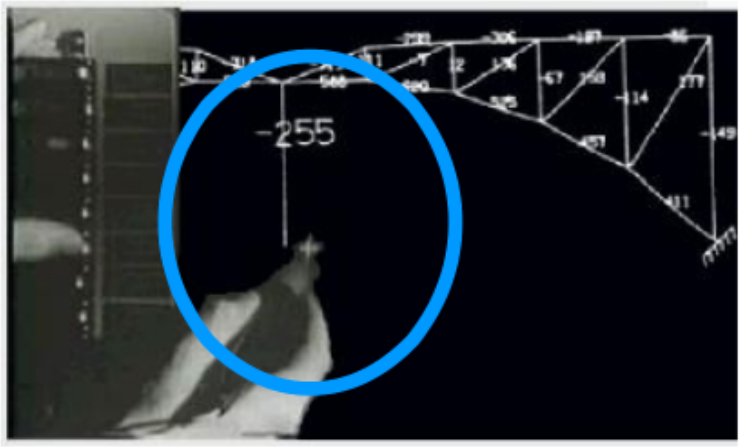


FIGURE 6.1. APPLYING TWO CONSTRAINTS INDIRECTLY TO TWO LINES

P PARALLELISM

play

stop

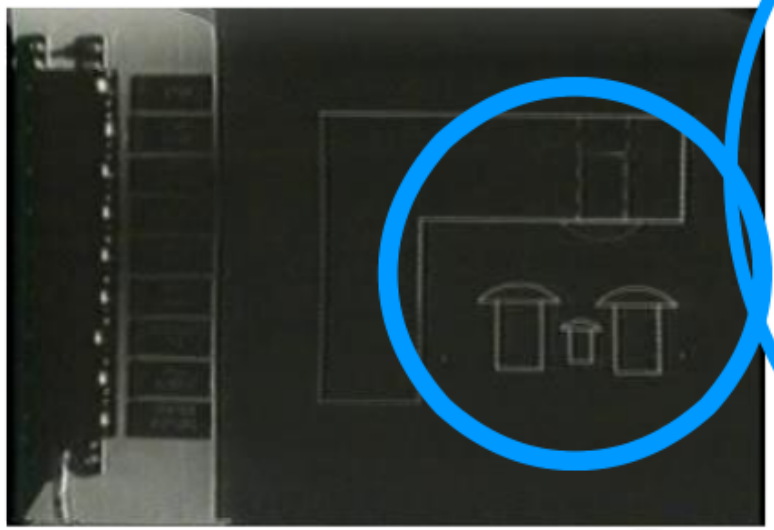


51st Birthday Of Sketchpad!

SKETCHPAD, A MAN-MACHINE GRAPHICAL COMMUNICATION SYSTEM

by

IVAN EDWARD SUTHERLAND



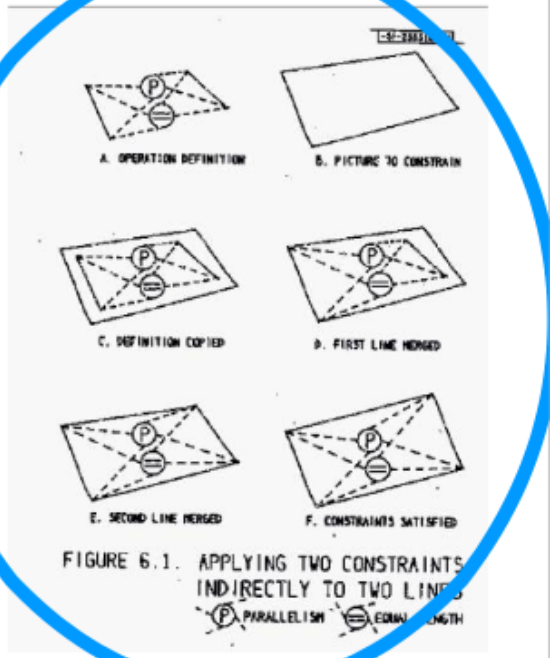
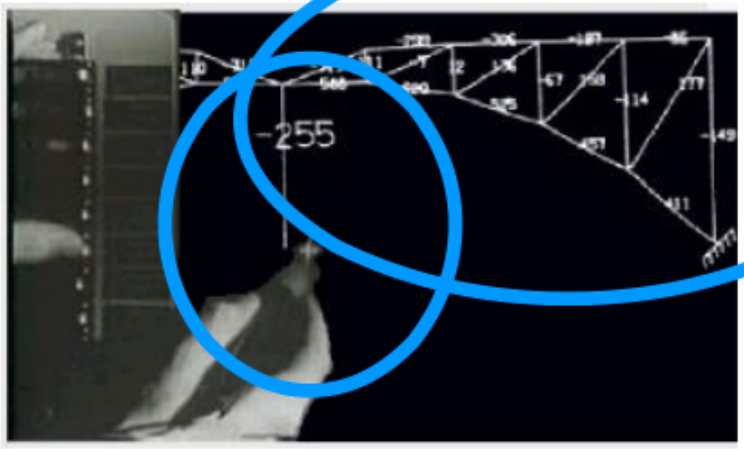
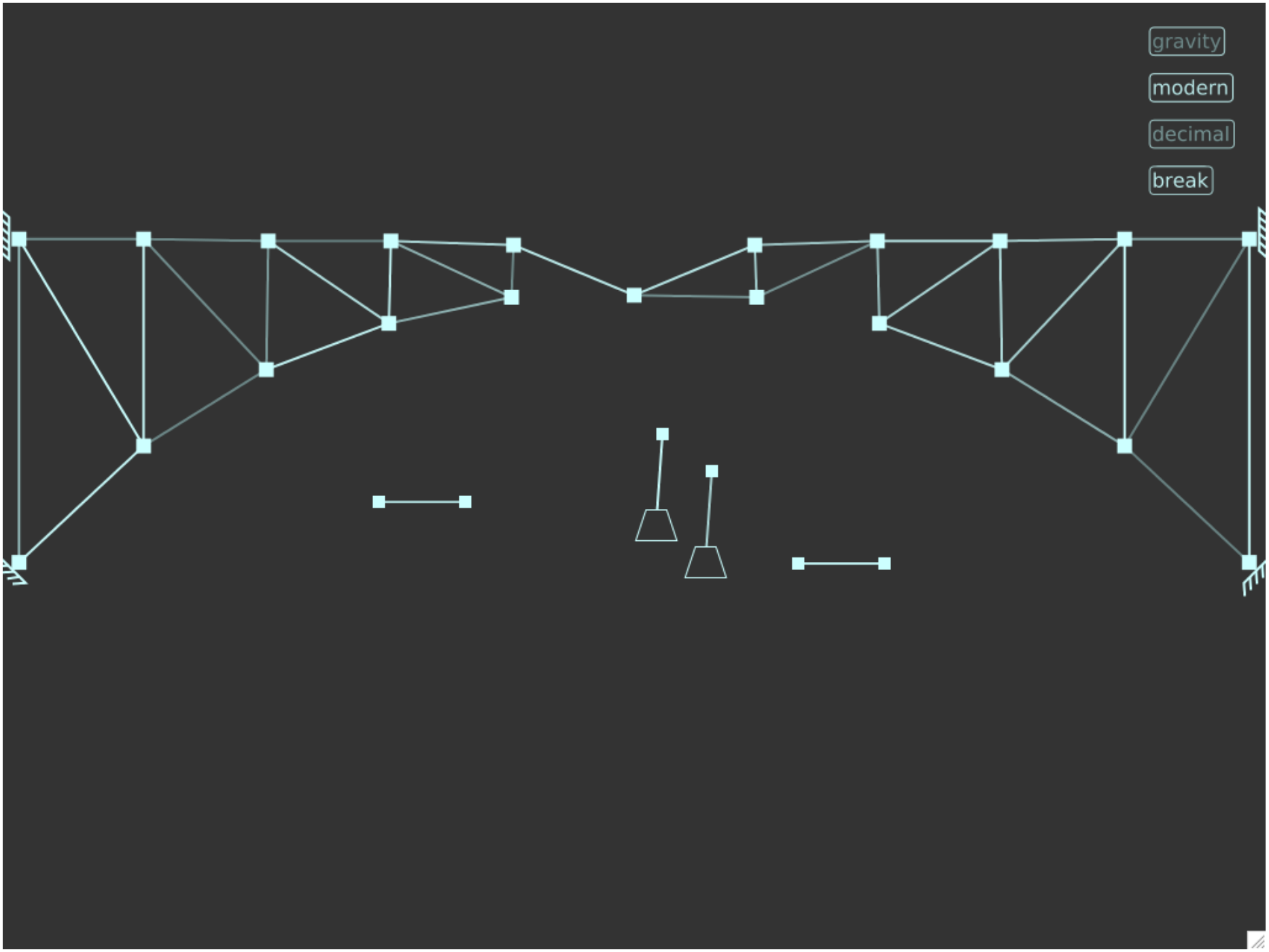


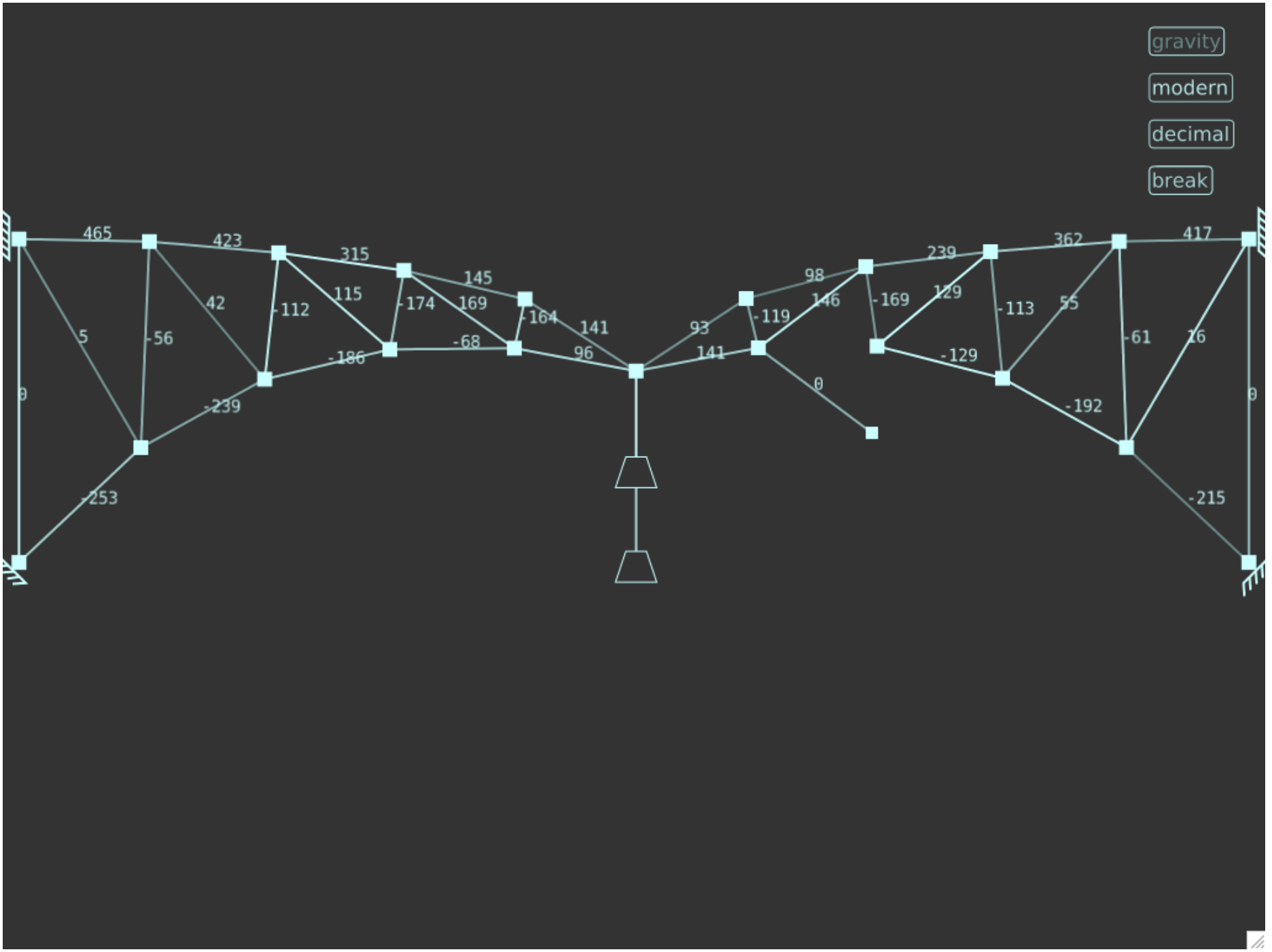
FIGURE 6.1. APPLYING TWO CONSTRAINTS INDIRECTLY TO TWO LINES

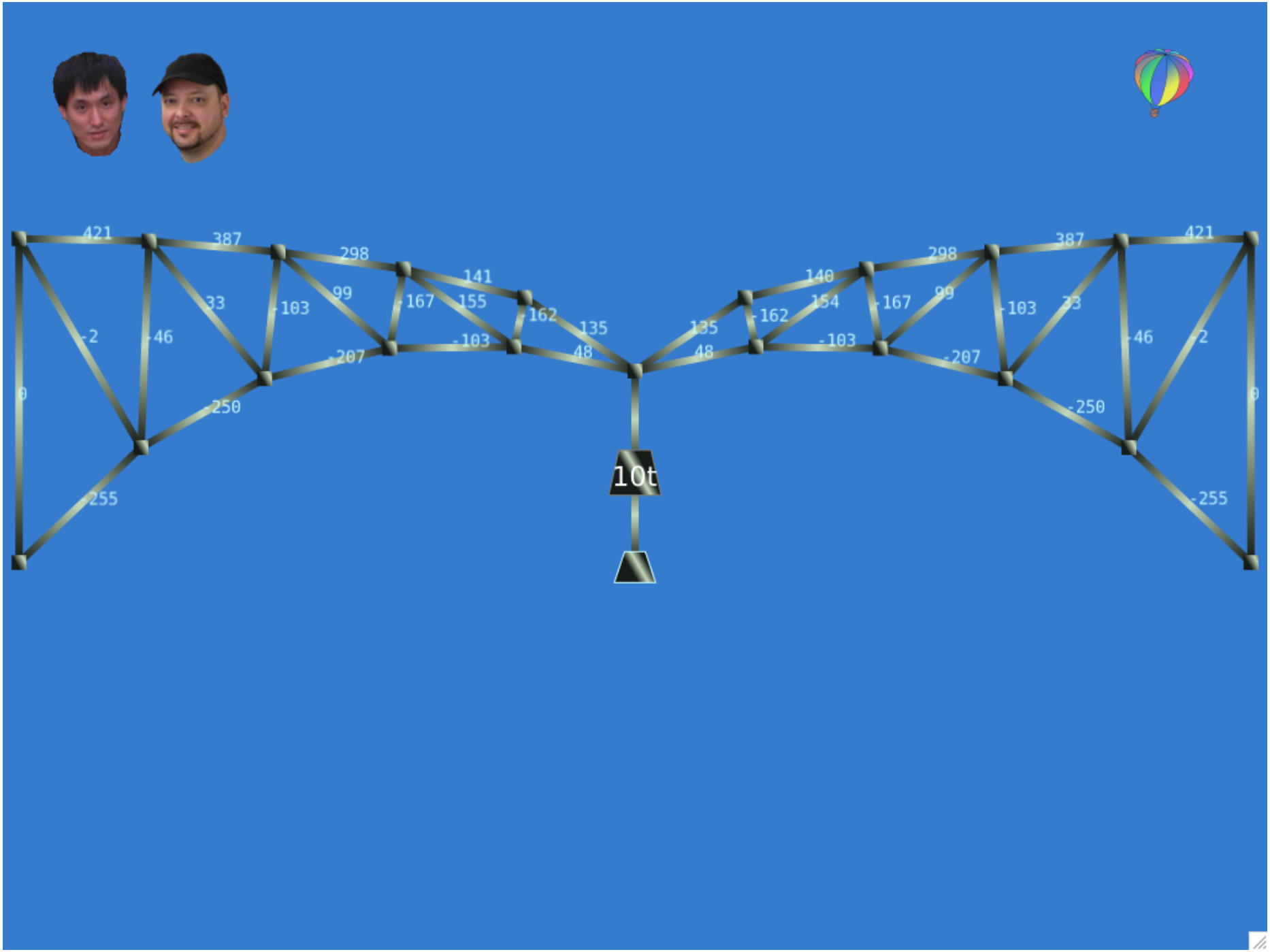
play

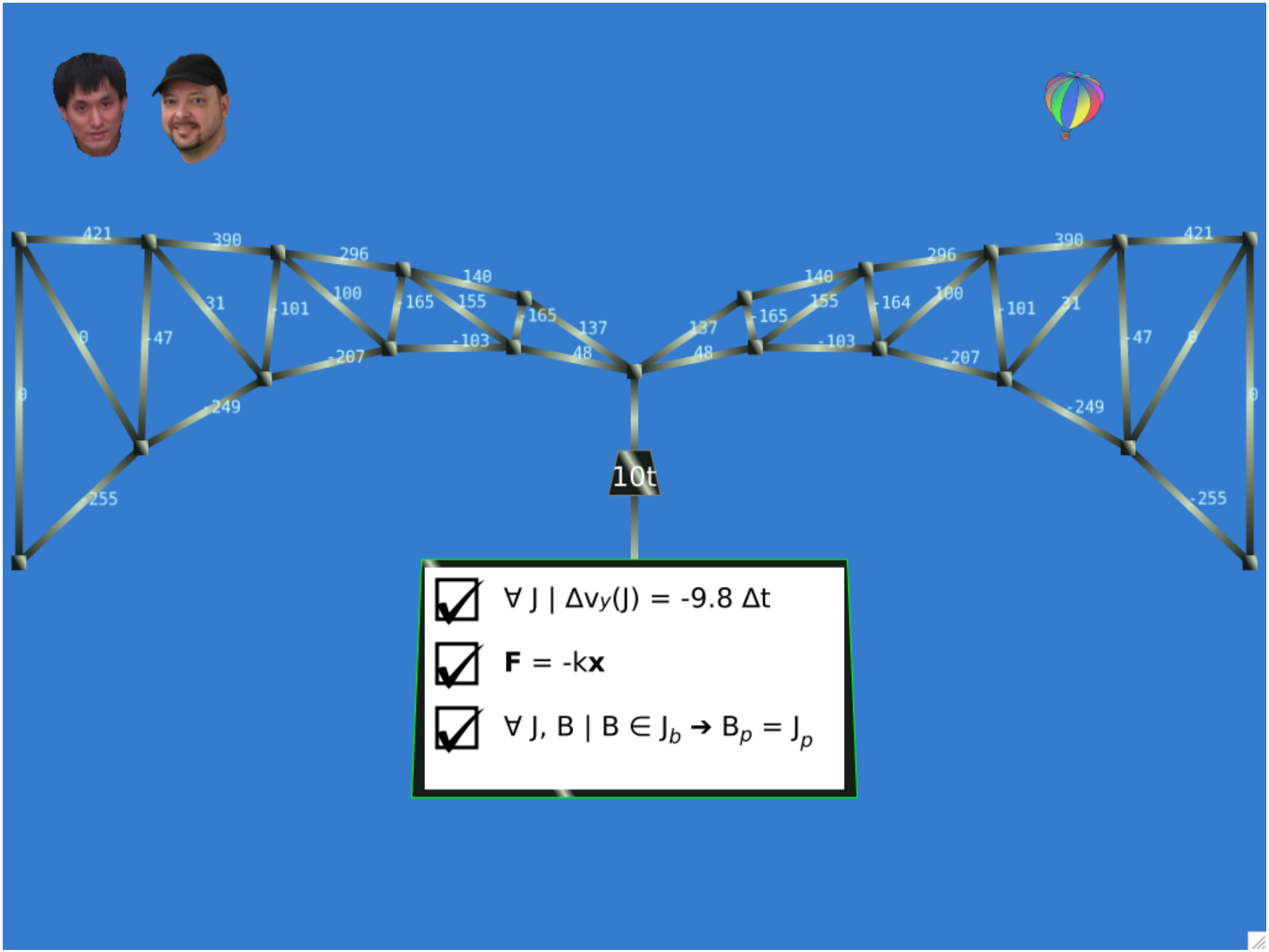
stop

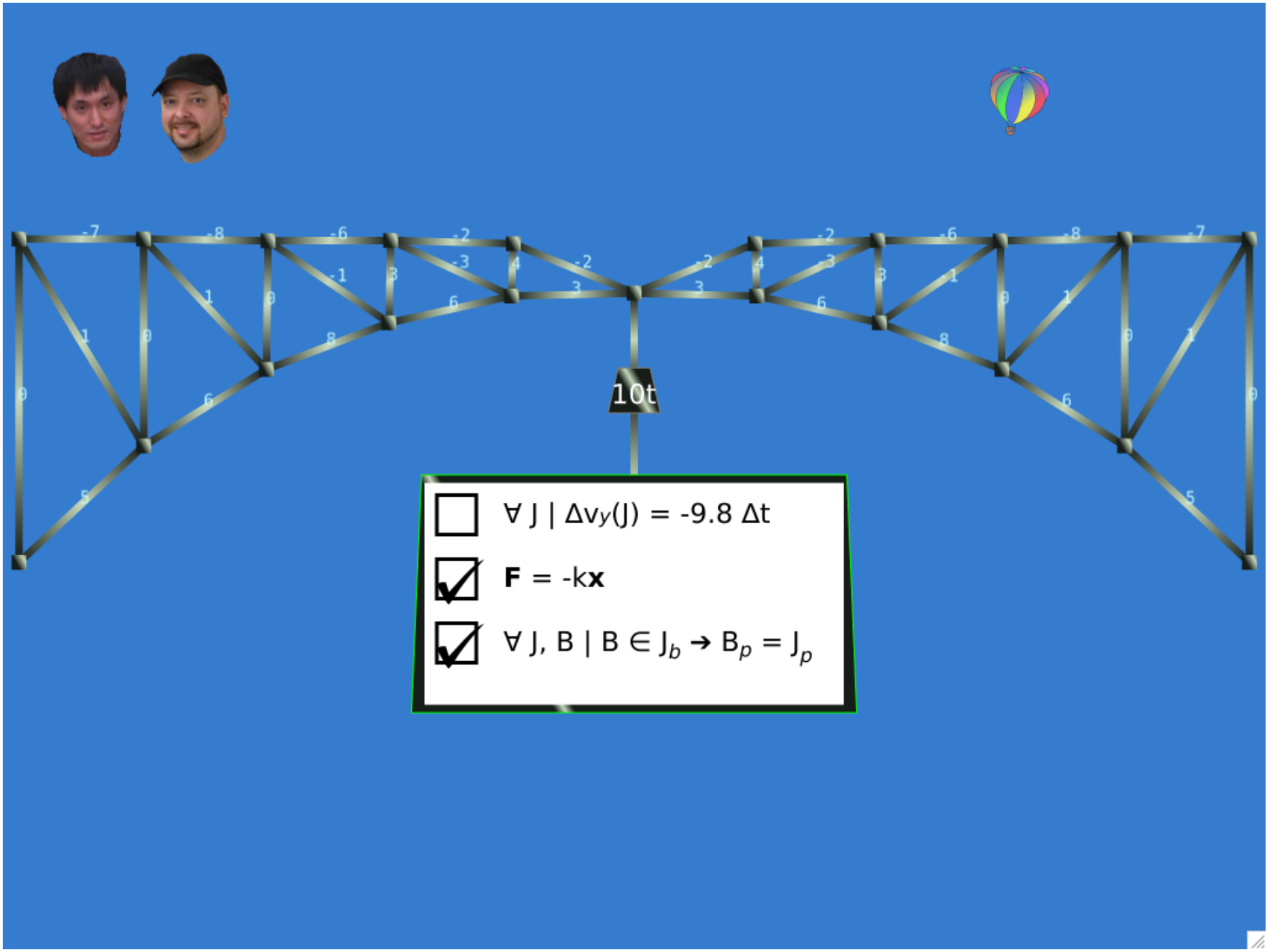


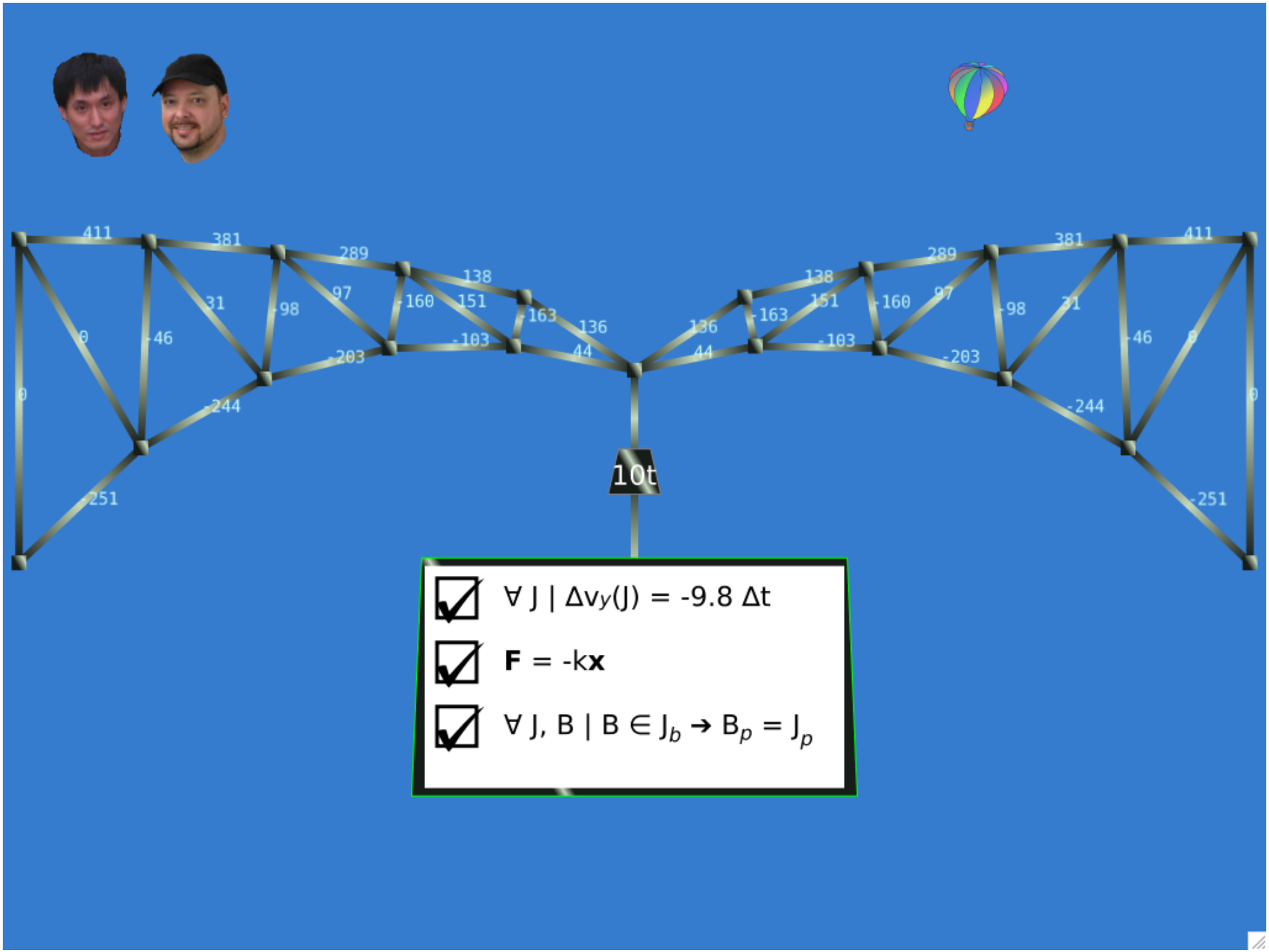

gravity
modern
decimal
break


gravity
modern
decimal
break
465
423
315
145
239
362
417
98
42
115
-112
-174
169
-164
141
93
-119
146
-169
129
-113
55
-61
16
5
-56
-68
96
141
-129
0
-186
-239
-192
0
0
-253
-215


421
387
298
141
140
298
387
421
33
103
99
167
155
162
135
135
162
154
167
99
103
33
2
46
103
207
48
48
103
207
46
2
0
250
250
0
10t
255
255

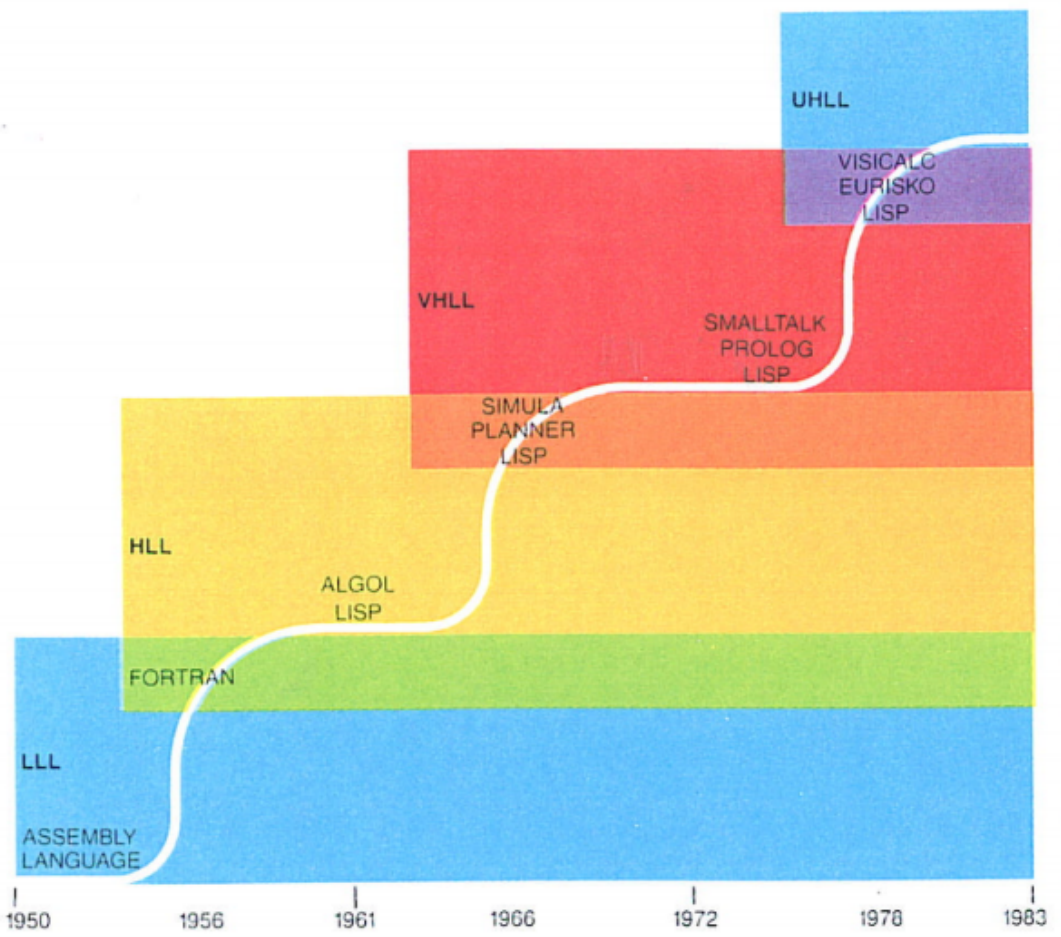

421
390
296
140
31
101
100
165
155
165
137
0
47
103
207
48
0
249
255
10t
☑ ∀ J | Δv_y(J) = -9.8 Δt
☑ F = -kx
☑ ∀ J, B | B ∈ J_b → B_p = J_p


10t
☐ ∀ J | Δv_y(J) = -9.8 Δt
☑ F = -kx
☑ ∀ J, B | B ∈ J_b → B_p = J_p


411
381
289
138
97
160
151
163
136
31
98
0
46
103
203
44
244
251
10t
☑ ∀ J | Δv_y(J) = -9.8 Δt
☑ F = -kx
☑ ∀ J, B | B ∈ J_b → B_p = J_p

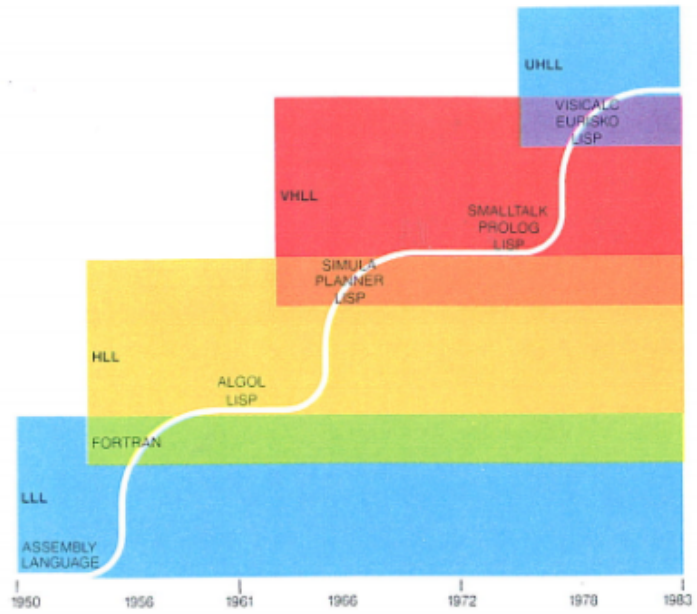
390
296
421
-165
-140
155
137
140
155
296
390
31
421
-47
0
0
31
-249
-255
-207
-101
100
-103
48
-137
-165
-202
-103
-101
-249
-47
0
-255

The “sun spot” theory of programming languages



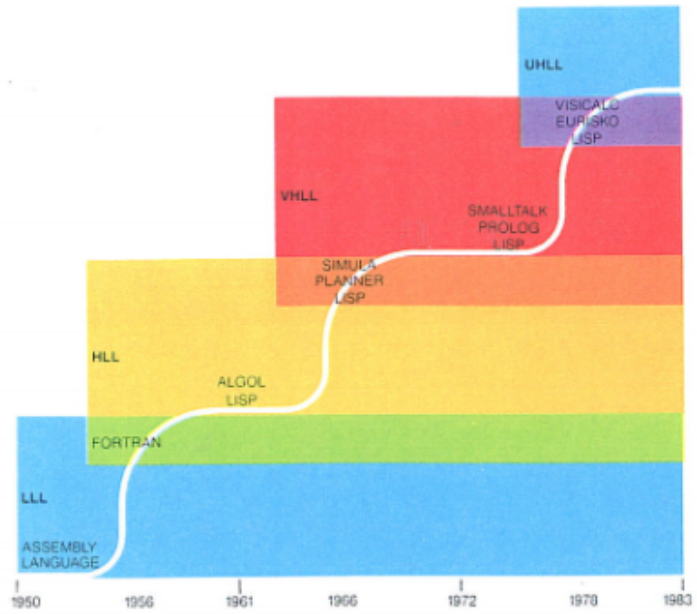
"Computer Software", Alan Kay, Scientific American Sep/1984

The “sun spot” theory of programming languages



"Computer Software", Alan Kay, Scientific American Sep/1984

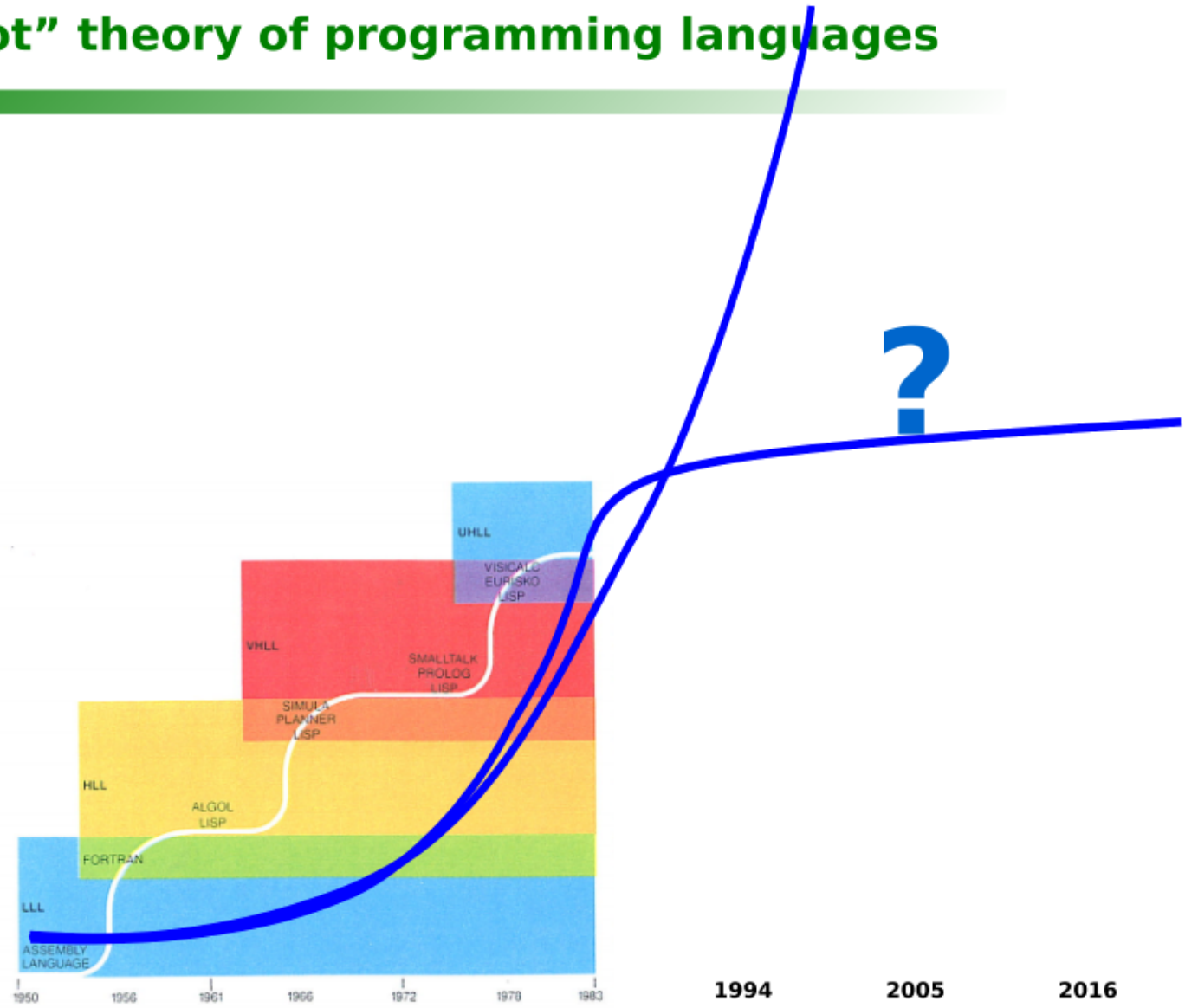
The “sun spot” theory of programming languages



1994 **2005** **2016**

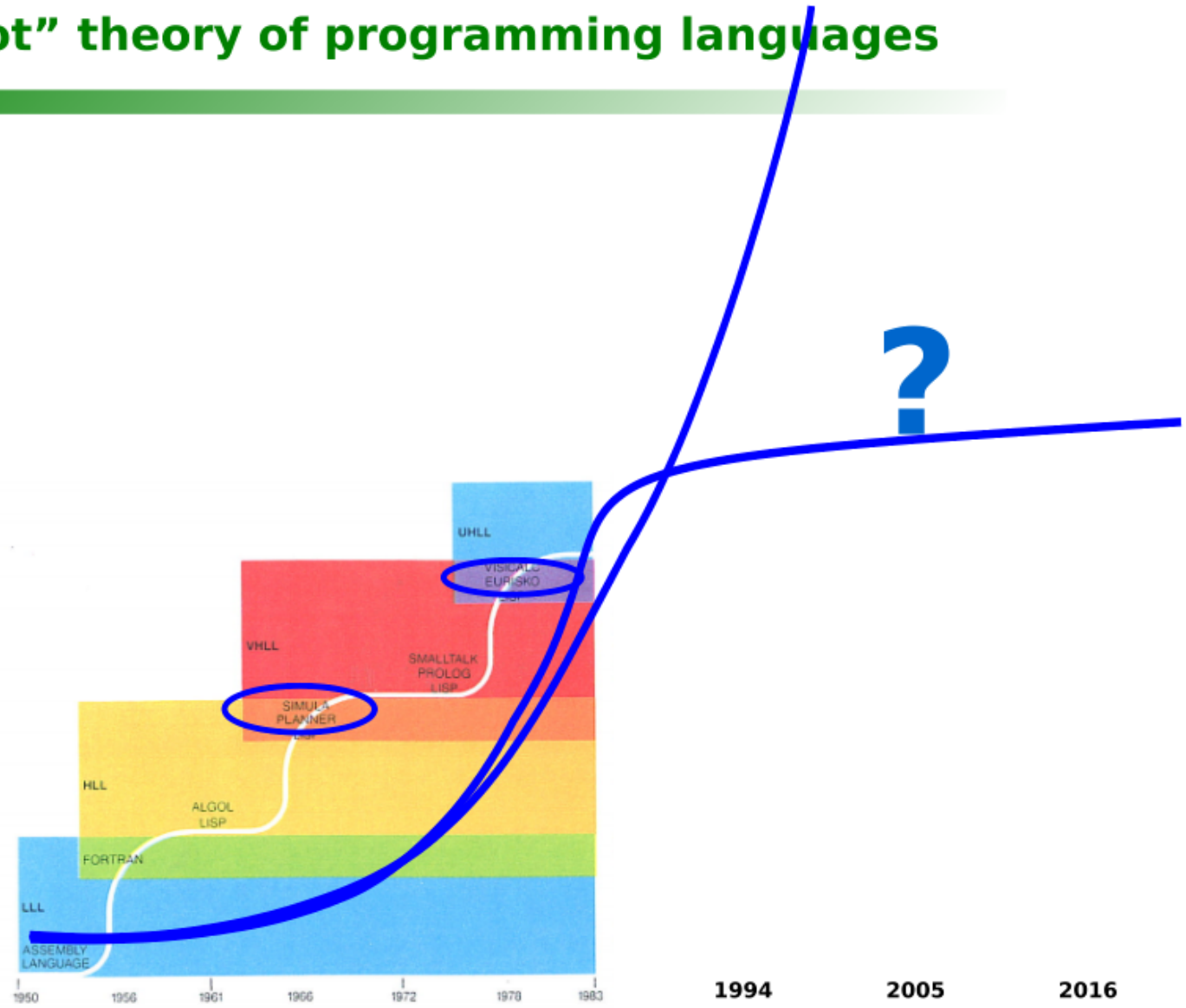
"Computer Software", Alan Kay, Scientific American Sep/1984

The “sun spot” theory of programming languages



"Computer Software", Alan Kay, Scientific American Sep/1984

The “sun spot” theory of programming languages



"Computer Software", Alan Kay, Scientific American Sep/1984

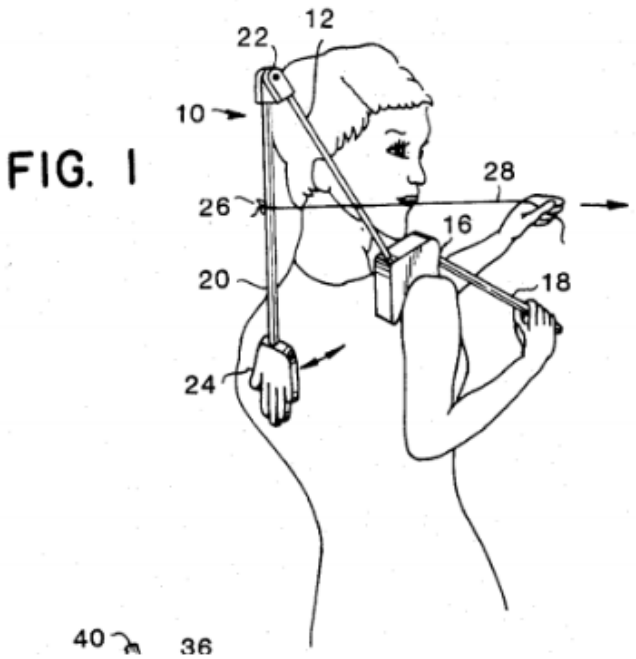
Further Reading/Watching Materials

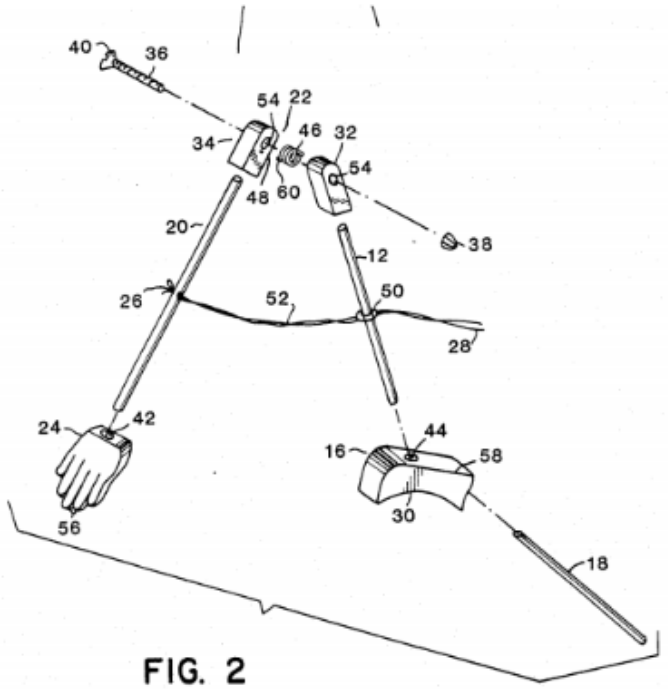
Viewpoints Research Institute Web:
Google: “VPRI Writings"

Alan Kay's recent talk:
Google: “Heidelberg Laureate Lecture Alan Kay"

Bret Victor's “Future of Programming” talk:
Google: “Future of Programmming Bret Victor”

Good Job!





"Pat on the back apparatus" (US Patent: 4,608,967, Sep 1986)