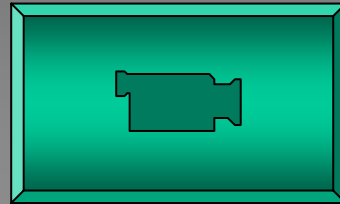


# Workshop on Single Particle Reconstructions And Visualization



# Scientific Animation

National Center for Macromolecular Imaging

December 14, 2002

[matthewd@bcm.tmc.edu](mailto:matthewd@bcm.tmc.edu)

# Scientific Animation

**Matthew T. Dougherty**

**National Center for  
Macromolecular Imaging**

**Baylor College of Medicine  
Houston, Texas**

*[matthewd@BCM.TMC.EDU](mailto:matthewd@BCM.TMC.EDU)*

# Scientific Animation

```
////////////////////////////////////
//
// Class: SbVec3f
//
// 3D vector used to represent points or directions. Each component of
// the vector is a floating-point number.
//
// WARNING!!!! Transcription of arrays of this class assume that the
// only data stored in this class are three consecutive values.
// Do not add any extra data members!!!
//
////////////////////////////////////

// C-api: prefix=SbV3f
// C-api.h: struct SbVec3f {
// C-api.h:         float vec[3];
// C-api.h: };

// C-api: end

class SbVec3f {
public:
// Default constructor
SbVec3f()

// Constructor given an array of 3 components
SbVec3f(const float v[3])

// Constructor given 3 individual components
SbVec3f(float x, float y, float z)

// Constructor given 3 planes
SbVec3f(SbPlane &p0, SbPlane &p1, SbPlane &p2);

// C-api: begin
// Returns right-handed cross product of vector and another vector
SbVec3f cross(const SbVec3f &v) const;
// C-api: end

// Returns dot (inner) product of vector and another vector
float dot(const SbVec3f &v) const;
// C-api.h: #define SbV3fDot(v0, v1)
// C-api.h: ((v0).vec[0] * (v1).vec[0] + (v0).vec[1] * (v1).vec[1] +
// C-api.h: (v0).vec[2] * (v1).vec[2])

// Returns pointer to array of 3 components
const float *getValue() const

// C-api.h: #define SbV3fGetXyz(xyz, src)
// C-api.h: (((xyz)[0] = (src).vec[0]), ((xyz)[1] = (src).vec[1]),
// C-api.h: ((xyz)[2] = (src).vec[2]))

// Returns 3 individual components
void getValue(float &x, float &y, float &z) const;
// C-api.h: #define SbV3fGetX_Y_Z(x, y, z, src)
// C-api.h: (((x) = (src).vec[0]), ((y) = (src).vec[1]), ((z) = (src).vec[2]))

// Returns geometric length of vector
float length() const;
// C-api.h: #define SbV3fLen(v)
// C-api.h: (sqrtf(SbV3fDot((v), (v))))

// C-api: begin
// Changes vector to be unit length
// C-api: name=norm
float normalize();
// C-api: end

// Returns 3 individual components
void getValue(float &x, float &y, float &z) const;
// C-api.h: #define SbV3fGetX_Y_Z(x, y, z, src)
// C-api.h: (((x) = (src).vec[0]), ((y) = (src).vec[1]), ((z) = (src).vec[2]))

// Returns geometric length of vector
float length() const;
// C-api.h: #define SbV3fLen(v)
// C-api.h: (sqrtf(SbV3fDot((v), (v))))

// C-api: begin
// Changes vector to be unit length
// C-api: name=norm
float normalize();
// C-api: end
```

```
const float *getValue() const
{ return vec; }
// C-api.h: #define SbV3fGetXyz(xyz, src)
// C-api.h: (((xyz)[0] = (src).vec[0]), ((xyz)[1] = (src).vec[1]),
// C-api.h: ((xyz)[2] = (src).vec[2]))

// Returns 3 individual components
void getValue(float &x, float &y, float &z) const;
// C-api.h: #define SbV3fGetX_Y_Z(x, y, z, src)
// C-api.h: (((x) = (src).vec[0]), ((y) = (src).vec[1]), ((z) = (src).vec[2]))

// Returns geometric length of vector
float length() const;
// C-api.h: #define SbV3fLen(v)
// C-api.h: (sqrtf(SbV3fDot((v), (v))))

// C-api: begin
// Changes vector to be unit length
// C-api: name=norm
float normalize();
// C-api: end

// Negates each component of vector in place
void negate();
// C-api.h: #define SbV3fNegate(v)
// C-api.h: SbV3fMultBy(v, -1.0)

// Sets value of vector from array of 3 components
SbVec3f & setValue(const float v[3])
// C-api.h: #define SbV3fSetXYZ(dest, src)
// C-api.h: (((dest).vec[0] = (src).vec[0]), ((dest).vec[1] = (src).vec[1]),
// C-api.h: ((dest).vec[2] = (src).vec[2]))

// Sets value of vector from 3 individual components
SbVec3f & setValue(float x, float y, float z)
// C-api.h: #define SbV3fSetX_Y_Z(dest, x, y, z)
// C-api.h: (((dest).vec[0] = (x)), ((dest).vec[1] = (y)),
// C-api.h: ((dest).vec[2] = (z)))

// Sets value of vector to be convex combination of 3 other
// vectors, using barycentric coordinates
SbVec3f & setV(const SbVec3f &v0, const SbVec3f &v1, const SbVec3f &v2);

// Accesses indexed component of vector
float & operator [(int i)]
const float & operator [(int i)] const

// Component-wise scalar multiplication and division operators
SbVec3f & operator *(float d);
// C-api.h: #define SbV3fMultBy(v, s)
// C-api.h: (((v).vec[0] *= (s)), ((v).vec[1] *= (s)), ((v).vec[2] *= (s)))

SbVec3f & operator /(float d)
{ return *this * (1.0 / d); }
// C-api.h: #define SbV3fDivBy(v, s)
// C-api.h: (((v).vec[0] /= (s)), ((v).vec[1] /= (s)), ((v).vec[2] /= (s)))

// Component-wise vector addition and subtraction operators
SbVec3f & operator +=(SbVec3f v);
SbVec3f & operator =(SbVec3f v);

// Nondestructive unary negation - returns a new vector
SbVec3f operator -(const SbVec3f &v);

// Component-wise binary scalar multiplication and division operators
friend SbVec3f operator *(const SbVec3f &v, float d);
friend SbVec3f operator *(float d, const SbVec3f &v)
{ return v * d; }
friend SbVec3f operator /(const SbVec3f &v, float d)
{ return v * (1.0 / d); }

// Component-wise binary vector addition and subtraction operators
friend SbVec3f operator +(const SbVec3f &v1, const SbVec3f &v2);
// C-api.h: #define SbV3fAdd(dest, src1, src2)
// C-api.h: (((dest).vec[0] = (src1).vec[0] + (src2).vec[0]),
// C-api.h: ((dest).vec[1] = (src1).vec[1] + (src2).vec[1]),
// C-api.h: ((dest).vec[2] = (src1).vec[2] + (src2).vec[2]))

friend SbVec3f operator -(const SbVec3f &v1, const SbVec3f &v2);
// C-api.h: #define SbV3fSub(dest, src1, src2)
// C-api.h: (((dest).vec[0] = (src1).vec[0] - (src2).vec[0]),
// C-api.h: ((dest).vec[1] = (src1).vec[1] - (src2).vec[1]),
// C-api.h: ((dest).vec[2] = (src1).vec[2] - (src2).vec[2]))

// Negates each component of vector in place
void negate();
// C-api.h: #define SbV3fNegate(v)
// C-api.h: SbV3fMultBy(v, -1.0)

// Sets value of vector from array of 3 components
SbVec3f & setValue(const float v[3])
{ vec[0] = v[0]; vec[1] = v[1]; vec[2] = v[2]; return *this; }
// C-api.h: #define SbV3fSetXYZ(dest, src)
```

```
const float *getValue() const
{ return vec; }
// C-api.h: #define SbV3fGetXyz(xyz, src)
// C-api.h: (((xyz)[0] = (src).vec[0]), ((xyz)[1] = (src).vec[1]),
// C-api.h: ((xyz)[2] = (src).vec[2]))

// Returns 3 individual components
void getValue(float &x, float &y, float &z) const;
// C-api.h: #define SbV3fGetX_Y_Z(x, y, z, src)
// C-api.h: (((x) = (src).vec[0]), ((y) = (src).vec[1]), ((z) = (src).vec[2]))

// Returns geometric length of vector
float length() const;
// C-api.h: #define SbV3fLen(v)
// C-api.h: (sqrtf(SbV3fDot((v), (v))))

// C-api: begin
// Changes vector to be unit length
// C-api: name=norm
float normalize();
// C-api: end

// Negates each component of vector in place
void negate();
// C-api.h: #define SbV3fNegate(v)
// C-api.h: SbV3fMultBy(v, -1.0)

// Sets value of vector from array of 3 components
SbVec3f & setValue(const float v[3])
{ vec[0] = v[0]; vec[1] = v[1]; vec[2] = v[2]; }
return *this; }
// C-api.h: #define SbV3fSetXYZ(dest, src)
// C-api.h: (((dest).vec[0] = (src)[0]), ((dest).vec[1] =
// C-api.h: (src)[1]),
// C-api.h: ((dest).vec[2] = (src)[2]))

// Sets value of vector from 3 individual components
SbVec3f & setValue(float x, float y, float z)
{ vec[0] = x; vec[1] = y; vec[2] = z; return *this; }
// C-api.h: #define SbV3fSetX_Y_Z(dest, x, y, z)
// C-api.h: (((dest).vec[0] = (x)), ((dest).vec[1] = (y)),
// C-api.h: ((dest).vec[2] = (z)))

// Sets value of vector to be convex combination of 3 other
// vectors, using barycentric coordinates
SbVec3f & setV(const SbVec3f &v0, const SbVec3f &v1, const SbVec3f &v2);

// Accesses indexed component of vector
float & operator [(int i)]
const float & operator [(int i)] const

// Component-wise scalar multiplication and division operators
SbVec3f & operator *(float d);
// C-api.h: #define SbV3fMultBy(v, s)
// C-api.h: (((v).vec[0] *= (s)), ((v).vec[1] *= (s)), ((v).vec[2] *= (s)))

SbVec3f & operator /(float d)
{ return *this * (1.0 / d); }
// C-api.h: #define SbV3fDivBy(v, s)
// C-api.h: (((v).vec[0] /= (s)), ((v).vec[1] /= (s)), ((v).vec[2] /= (s)))

// Component-wise vector addition and subtraction operators
SbVec3f & operator +=(SbVec3f v);
SbVec3f & operator =(SbVec3f v);

// Nondestructive unary negation - returns a new vector
SbVec3f operator -(const SbVec3f &v);

// Component-wise binary scalar multiplication and division operators
friend SbVec3f operator *(const SbVec3f &v, float d);
friend SbVec3f operator *(float d, const SbVec3f &v)
{ return v * d; }
friend SbVec3f operator /(const SbVec3f &v, float d)
{ return v * (1.0 / d); }

// Component-wise binary vector addition and subtraction operators
friend SbVec3f operator +(const SbVec3f &v1, const SbVec3f &v2);
// C-api.h: #define SbV3fAdd(dest, src1, src2)
// C-api.h: (((dest).vec[0] = (src1).vec[0] + (src2).vec[0]),
// C-api.h: ((dest).vec[1] = (src1).vec[1] + (src2).vec[1]),
// C-api.h: ((dest).vec[2] = (src1).vec[2] + (src2).vec[2]))

friend SbVec3f operator -(const SbVec3f &v1, const SbVec3f &v2);
// C-api.h: #define SbV3fSub(dest, src1, src2)
// C-api.h: (((dest).vec[0] = (src1).vec[0] - (src2).vec[0]),
// C-api.h: ((dest).vec[1] = (src1).vec[1] - (src2).vec[1]),
// C-api.h: ((dest).vec[2] = (src1).vec[2] - (src2).vec[2]))

// Negates each component of vector in place
void negate();
// C-api.h: #define SbV3fNegate(v)
// C-api.h: SbV3fMultBy(v, -1.0)

// Sets value of vector from array of 3 components
SbVec3f & setValue(const float v[3])
{ vec[0] = v[0]; vec[1] = v[1]; vec[2] = v[2]; return *this; }
// C-api.h: #define SbV3fSetXYZ(dest, src)
```

# Scientific Animation

*Lecture Lite*

# Scientific Animation

## *PROS*

- ✓ Conveys 3D well
- ✓ Storytelling
- ✓ Engages audience
- ✓ Essential for dynamics

# Scientific Animation

## *PROS*

- ✓ Conveys 3D well
- ✓ Storytelling
- ✓ Engages audience
- ✓ Essential for dynamics

## *CONS*

- ☠ Easily abused
- ☠ Entertainment
- ☠ Gimmick

# Scientific Animation

## *RECIPE*

- 1) establish scientific constraints
- 2) create multi-dimensional visualization
- 3) generate sequence of 2D images
- 4) playback at 24 fps or greater

# **Scientific Animation**

*Lecture Long*

# Scientific Animation

- History & philosophy

# Scientific Animation

- History & philosophy
- Methodologies

# Scientific Animation

- History & philosophy
- Methodologies
- Examples

# History

- Visual symbolism 50,000 bce/Australia

# History

- Visual symbolism 50,000 bce/Australia
- Color graphics 30,000 bce/France

# History

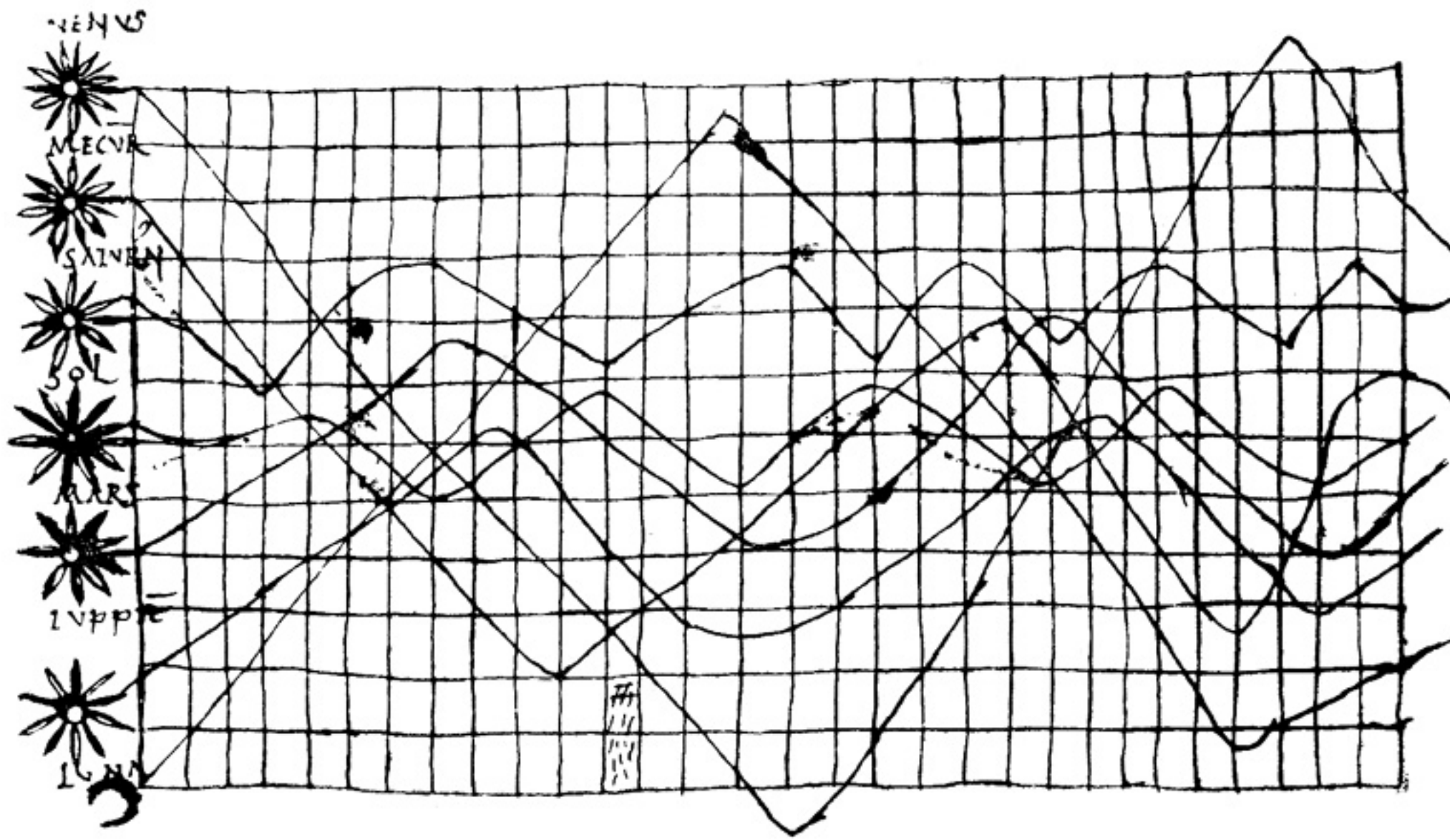
- Visual symbolism 50,000 bce/Australia
- Color graphics 30,000 bce/France
- Geographic Map 2300 bce/Babylon

# History

- Visual symbolism 50,000 bce/Australia
- Color graphics 30,000 bce/France
- Geographic Map 2300 bce/Babylon
- Geometric abstraction 300 bce/Euclid

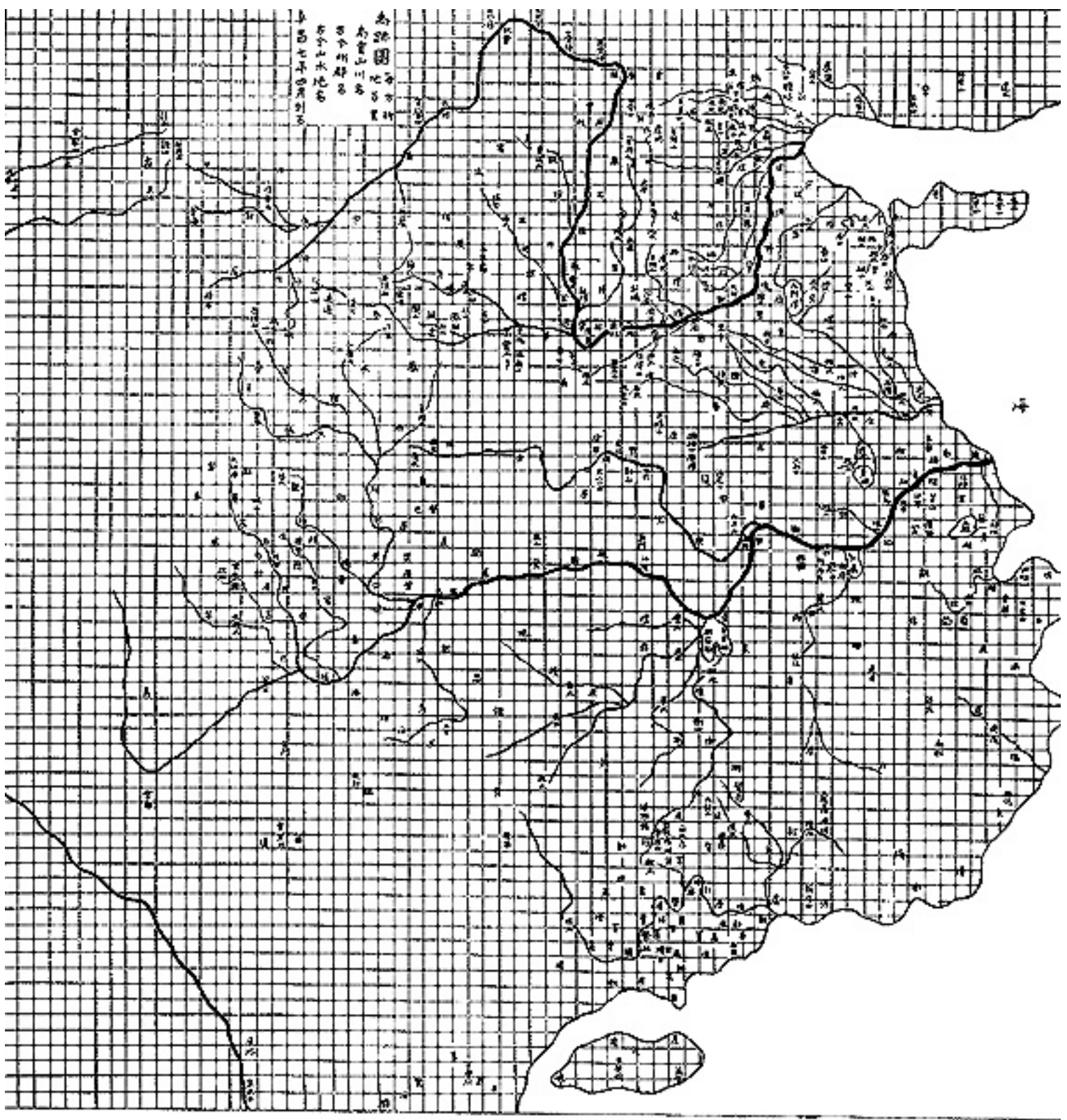
# History

- Visual symbolism 50,000 bce/Australia
- Color graphics 30,000 bce/France
- Geographic Map 2300 bce/Babylon
- Geometry 300 bce/Euclid
- Time series 1000/astronomy



# History

- Visual symbolism 50,000 bce/Australia
- Color graphics 30,000 bce/France
- Geographic Map 2300 bce/Babylon
- Geometry 300 bce/Euclid
- Time series 1000/astronomy
- Geographic metric 1100/China

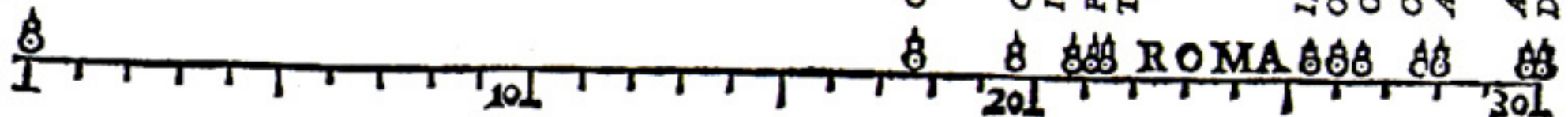


# History

- Visual symbolism 50,000 bc/Australia
- Color graphics 30,000 bc/France
- Geographic Map 2300 bc/Babylon
- Geometry 300 bc/Euclid
- Time series 1000/astronomy
- Geographic metric 1100/China
- 1D statistical 1644/van Langren

TOLEDO.

GRADOS DE LA LONGITUD.



G. Iansonius.

G. Mercator.

I. Schonerus.

P. Lantsbergius.

T. Brab.

L. Regiomontanus.

Oronius.

C. Clavius.

C. Ptolomæus.

A. Argelæus.

A. Maginus.

D. Origanus.

ROMA

10

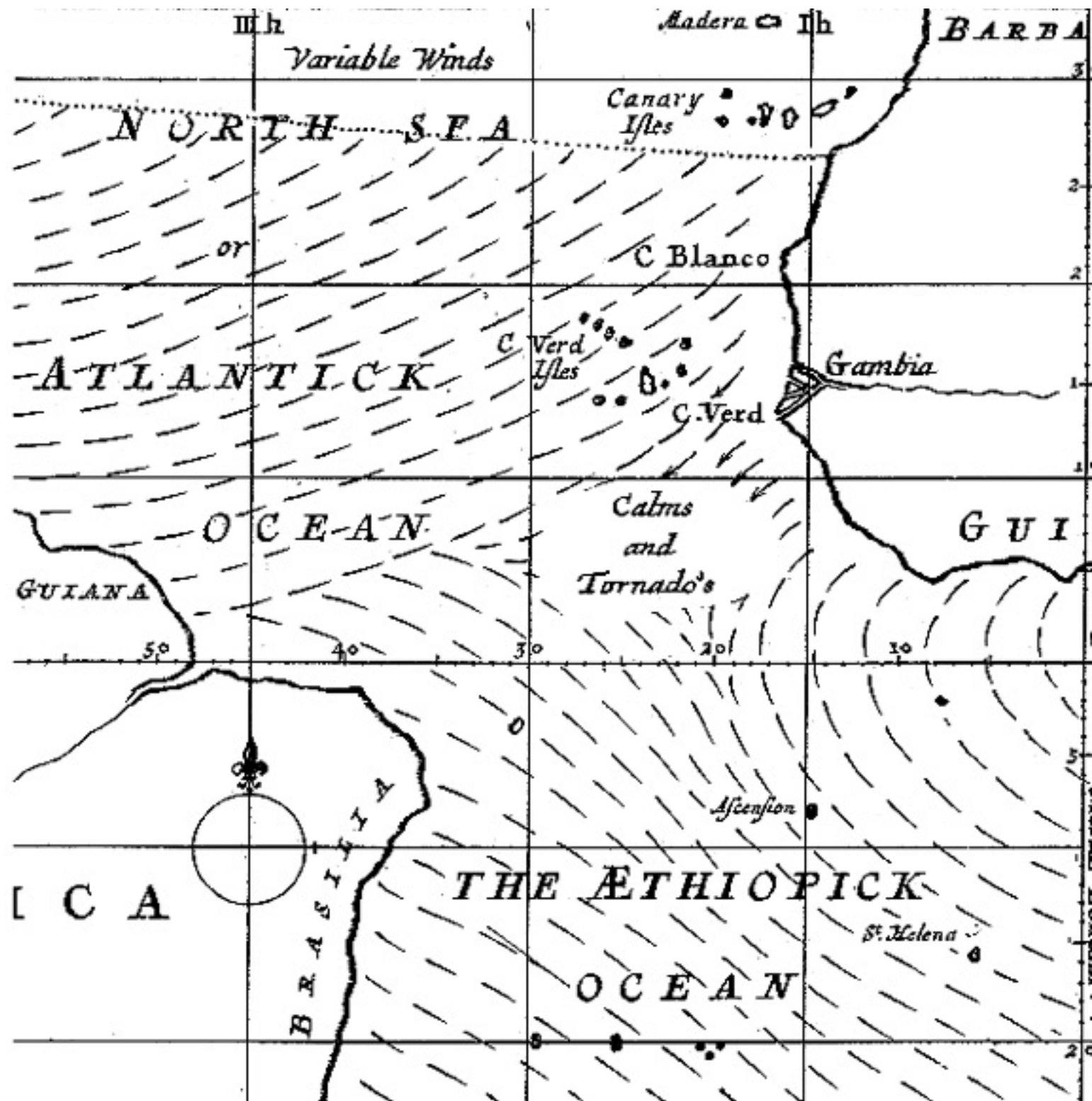
20

30



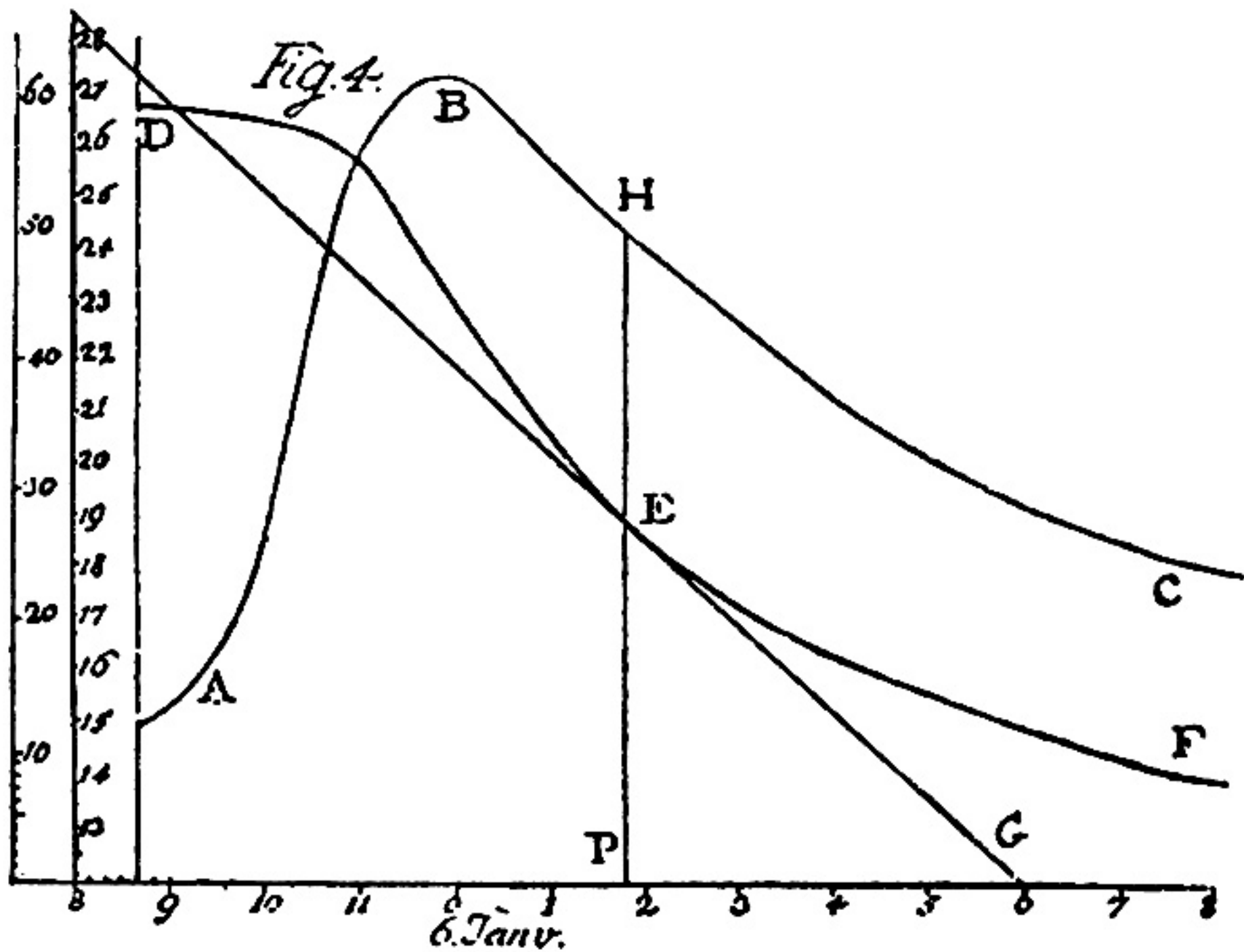
# History

- Visual symbolism 50,000 bce/Australia
- Color graphics 30,000 bce/France
- Geographic Map 2300 bce/Babylon
- Geometry 300 bce/Euclid
- Time series 1000/astronomy
- Geographic metric 1100/China
- 1D statistical 1644/Langren
- Thematic map 1686/Halley



# History

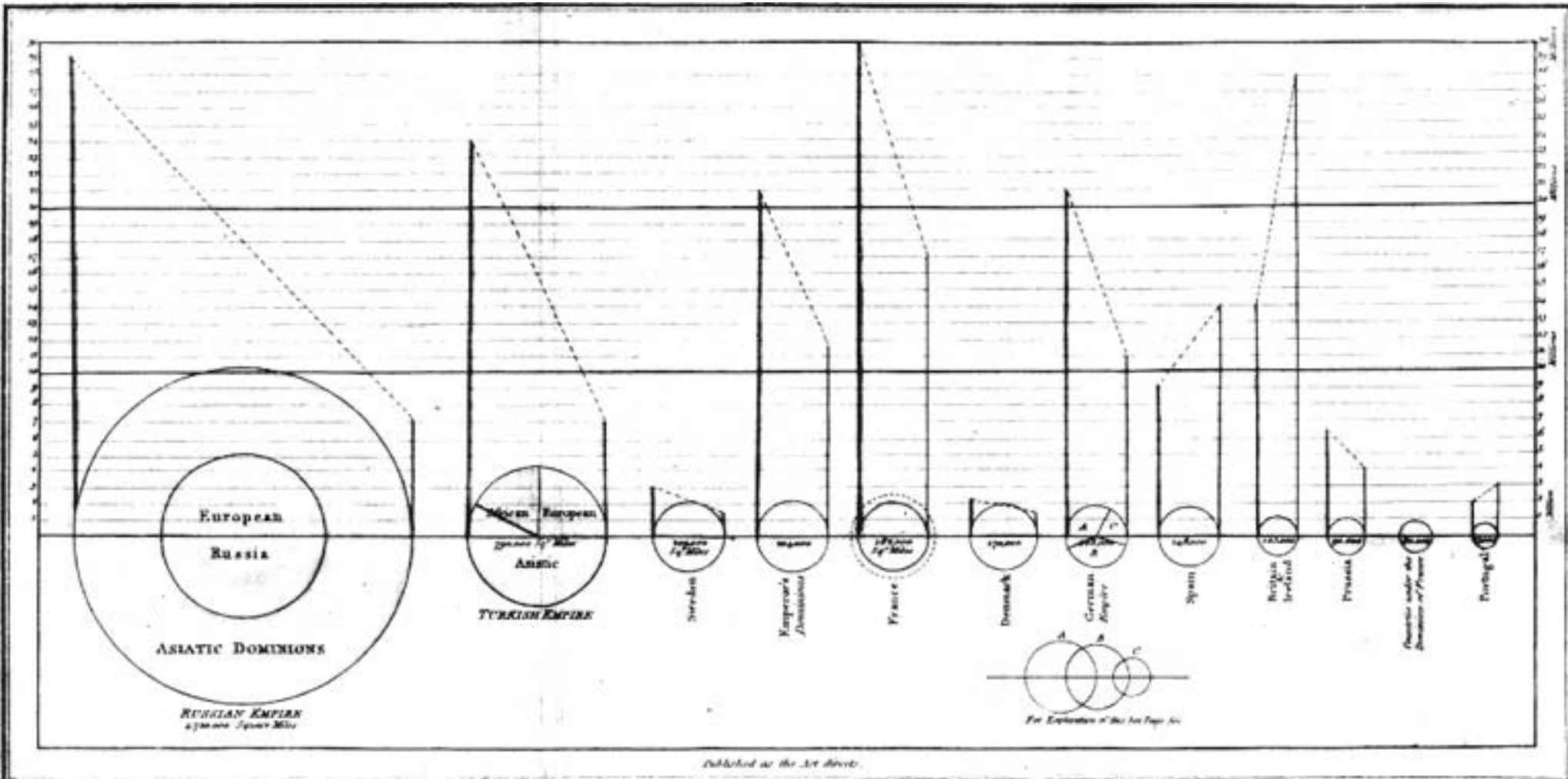
- Visual symbolism 50,000 bce/Australia
- Color graphics 30,000 bce/France
- Geographic Map 2300 bce/Babylon
- Geometry 300 bce/Euclid
- Time series 1000/astronomy
- Geographic metric 1100/China
- 1D statistical 1644/Langren
- Thematic map 1686/Halley
- 2D abstraction 1765/Lambert



# History

- nD abstraction

1801/Playfair



# History

- nD abstraction 1801/Playfair
- Time & vision 1824/Roget

# History

- nD abstraction 1801/Playfair
- Time & vision 1824/Roget
- Scientific photography 1872/Muybridge

# History

- nD abstraction 1801/Playfair
- Time & vision 1824/Roget
- Scientific photography 1872/Muybridge
- Animation 1906/Blackton

# History

- nD abstraction 1801/Playfair
- Time & vision 1824/Roget
- Scientific photography 1872/Muybridge
- Animation 1906/Blackton
- **Computer graphics** 1950's/MIT

# History

- nD abstraction 1801/Playfair
- Time & vision 1824/Roget
- Scientific photography 1872/Muybridge
- Animation 1906/Blackton
- Computer graphics 1950's/MIT
- Computer animation 1960's

# History

- nD abstraction 1801/Playfair
- Time & vision 1824/Roget
- Scientific photography 1872/Muybridge
- Animation 1906/Blackton
- Computer graphics 1950's/MIT
- Computer animation 1960's
- **Rendering & modeling 1970's**

# History

- nD abstraction 1801/Playfair
- Time & vision 1824/Roget
- Scientific photography 1872/Muybridge
- Animation 1906/Blackton
- Computer graphics 1950's/MIT
- Computer animation 1960's
- Rendering & modeling 1970's
- Animation applications 1980's

# History

- nD abstraction 1801/Playfair
- Time & vision 1824/Roget
- Scientific photography 1872/Muybridge
- Animation 1906/Blackton
- Computer graphics 1950's/MIT
- Computer animation 1960's
- Rendering & modeling 1970's
- Animation applications 1980's
- Affordable graphics 1990's

# Philosophy

# Epistemology

“Notions of space are rooted in our physiological organism. Geometric concepts are the product of the idealization of physical experiences of space. Systems of geometry, finally, originate in the logical classification of the conceptual materials so obtained.

“Epistemological inquires regarding space and geometry accordingly concern the physiologist, the psychologist, the physicist, the mathematician, the philosopher, *{the artist,}* and logician alike, and they can be gradually carried to their definitive solution only by the consideration of the widely disparate points of view ...”

Ernst Mach

# Philosophical cornerstones

- The mathematical discipline of geometry used as a means to describe relations of physical experience.

# Philosophical cornerstones

- The mathematical discipline of geometry used as a means to describe relations of physical experience.
- Knowledge is frequently equated to thoughtful observation, although understanding and the sensation of visualization are not equal.

# Philosophical cornerstones

- The mathematical discipline of geometry used as a means to describe relations of physical experience.
- Knowledge is frequently equated to thoughtful observation, although understanding and the sensation of visualization are not equal.
- Data is a measurement described by a number.

# Philosophical Issues

# Philosophical Issues

- Presenter versus audience

# Philosophical Issues

- Presenter versus audience
- Disinformation versus information

# Philosophical Issues

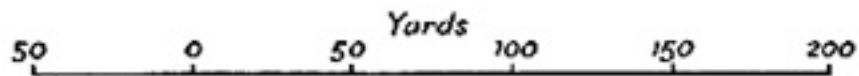
- Presenter versus audience
- Disinformation versus information

# Philosophical Issues

- Presenter versus audience
- Disinformation versus information
- Illusion versus illustration
- Complexity versus simplicity

# Philosophical Issues

- Presenter versus audience
- Disinformation versus information
- Illusion versus illustration
- Complexity versus simplicity
- Example: Cholera Epidemic, 1854



X Pump    • Deaths from cholera



# Philosophical Values

- Presenter versus audience
- Disinformation versus information
- Illusion versus illustration
- Complexity versus simplicity
- Example: Cholera Epidemic, 1854
- Example: Challenger Launch, 1986

HISTORY OF O-RING DAMAGE ON SRM FIELD JOINTS

1161  
OCT 30, 1985

61A LH Center Field\*\*  
61A LH CENTER FIELD\*\*  
51C LH Forward Field\*\*  
51C RH Center Field (prim)\*\*\*  
51C RH Center Field (sec)\*\*\*

SRM No.	Cross Sectional View			Top View		Clocking Location (deg)	
	Erosion Depth (in.)	Perimeter Affected (deg)	Nominal Dia. (in.)	Length Of Max Erosion (in.)	Total Heat Affected Length (in.)		
22A	None	None	0.280	None	None	36° -- 66°	
22A	NONE	NONE	0.280	NONE	NONE	338° -- 18°	
15A	0.010	154.0	0.280	4.25	5.25	163	
15B	0.038	130.0	0.280	12.50	58.75	354	
15B	None	45.0	0.280	None	29.50	354	
41D RH Forward Field	13B	0.028	110.0	0.280	3.00	None	275
41C LH Aft Field*	11A	None	None	0.280	None	None	--
41B LH Forward Field	10A	0.040	217.0	0.280	3.00	14.50	351
STS-2 RH Aft Field	2B	0.053	116.0	0.280	--	--	90

\*Hot gas path detected in putty. Indication of heat on O-ring, but no damage.

\*\*Soot behind primary O-ring.

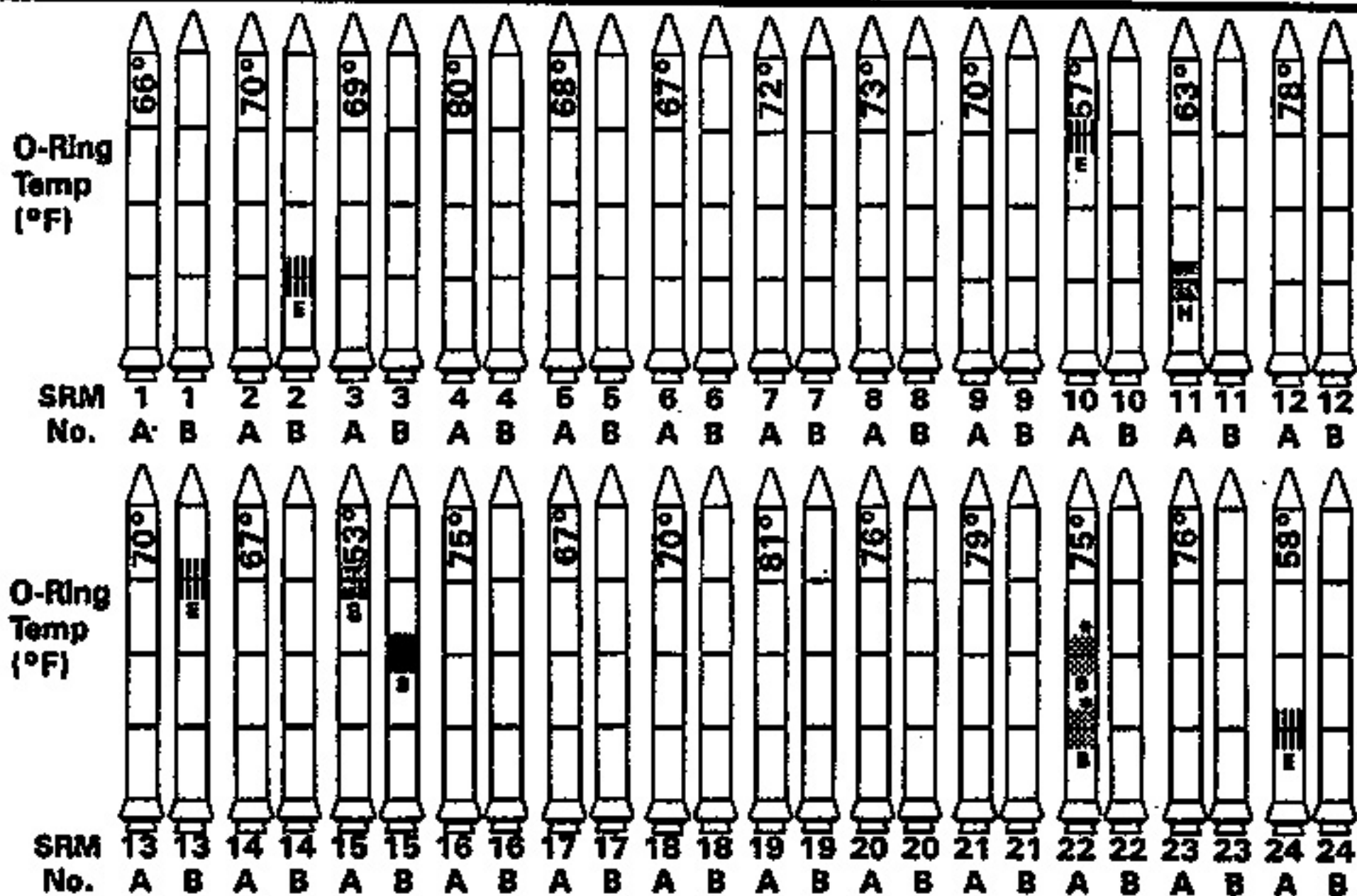
\*\*\*Soot behind primary O-ring, heat affected secondary O-ring.

Clocking location of leak check port - 0 deg.

OTHER SRM-15 FIELD JOINTS HAD NO BLOWHOLES IN PUTTY AND NO SOOT NEAR OR BEYOND THE PRIMARY O-RING.

SRM-22 FORWARD FIELD JOINT HAD PUTTY PATH TO PRIMARY O-RING, BUT NO O-RING EROSION AND NO SOOT BLOWBY. OTHER SRM-22 FIELD JOINTS HAD NO BLOWHOLES IN PUTTY.

# History of O-Ring Damage In Field Joints (Cont)



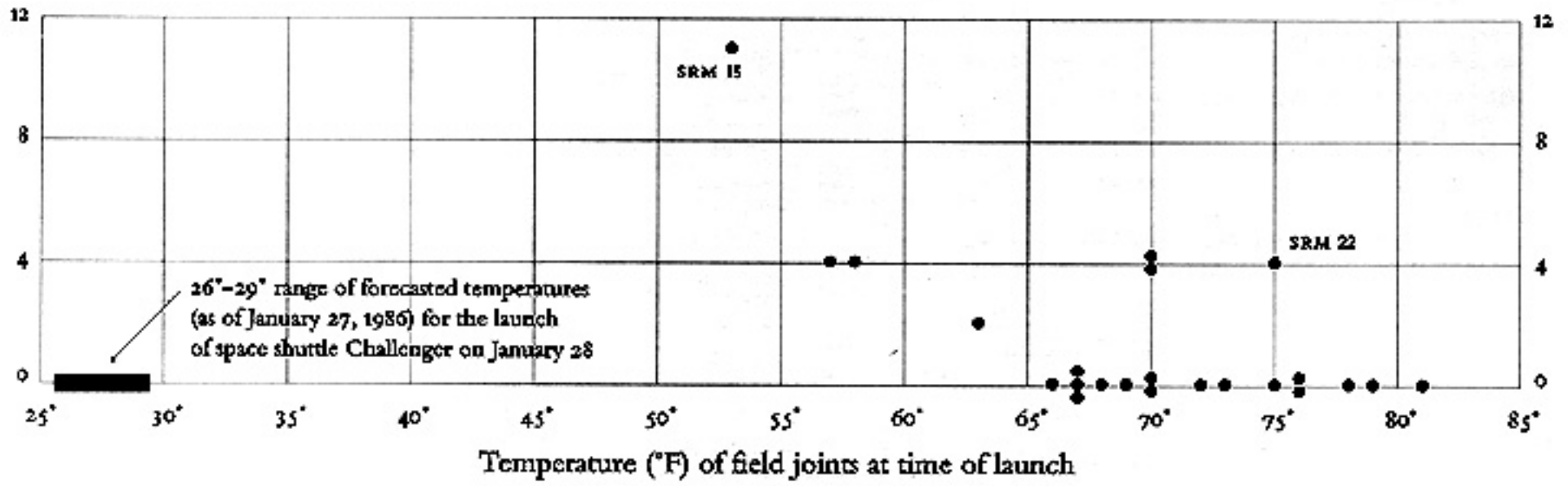
MORTON THICKOL, INC.  
Watch Operations

\* No Erosion

0000-11

INFORMATION ON THIS PAGE WAS PREPARED TO SUPPORT AN ORAL PRESENTATION AND CANNOT BE CONSIDERED COMPLETE WITHOUT THE ORAL DISCUSSION

O-ring damage index, each launch



Tufte, 1997

- Lies
- Damnable lies
- Statistics

# Mark Twain

- Lies
- Damnable lies
- Statistics

## Matt's update

- Visualization
- Animation

# Methodologies

# Methodologies

- Principles of computer graphics

# Methodologies

- Principles of computer graphics
- Principles of scientific visualization

# Methodologies

- Principles of computer graphics
- Principles of scientific visualization
- General design guidelines

# Methodologies

- Principles of computer graphics
- Principles of scientific visualization
- General design guidelines
- Tufte's view of graphical excellence

# Methodologies

- Principles of computer graphics
- Principles of scientific visualization
- General design guidelines
- Tufte's view of graphical excellence
- Kellers' effective visualization

# Methodologies

- Principles of computer graphics
- Principles of scientific visualization
- General design guidelines
- Tufte's view of graphical excellence
- Kellers' effective visualization
- Storytelling with animation

# Methodologies

- Principles of computer graphics
- Principles of scientific visualization
- General design guidelines
- Tufte's view of graphical excellence
- Kellers' effective visualization
- Storytelling with animation
- Living within a budget

# Principles of computer

- Graphics, a subset of computing

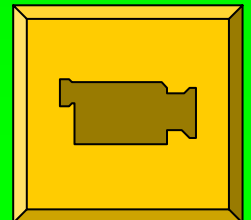
# Principles of computer

- Graphics, a subset of computing
- Space, Camera, & Coordinates
- Color
- Images & Rendering
- Output media
- Animation

# Principles of computer

## Space, Camera, & Coordinates

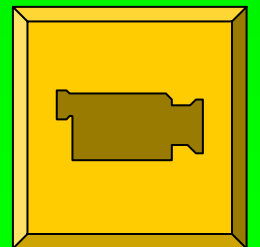
- Space: pixels, polygons, or volumes
- Camera: cone of vision
- Objects populate the space
- Background & Lighting
- Coordinates: translation & orientation
- Coordinates: world, camera, objects



# Principles of computer

## Color

- Color space: RGB, HSV, CYMK
- Color model: diffuse, specular, ambient, emissive
- Transparency



# Principles of computer

## Color

- Color space: RGB, HSV, CYMK
- Color model: diffuse, specular, ambient, emissive
- Transparency
- Color schemes
- Physiological considerations
- Cultural considerations

# Principles of computer graphics

## Images & Rendering

- 2D image formats: RGB, RGBA
- Rendering: Process of 3D to 2D conversion
- Space, camera, & lighting
- Annotations
- Overlays & composites
- Simple algorithms versus ray tracing

# Principles of computer

## Output media

- Paper & Film
- Analog video: TV
- Computer displays
- Projectors
- Image files: TIFF, GIF, JPEG
- Digital video: ISO-MPEG, MS-AVI

# Principles of computer

## Animation

- Sequence of images referenced in time
- 24,25,30 fps

# Principles of scientific

- Exploration, Analysis, Presentation

# Principles of scientific

- Exploration, Analysis, Presentation
- Determine format of data

# Principles of scientific

- Exploration, Analysis, Presentation
- Determine format of data
- Determine size of data

# Principles of scientific

- Exploration, Analysis, Presentation
- Determine format of data
- Determine size of data
- Determine computational capabilities

# Principles of scientific

- Exploration, Analysis, Presentation
- Determine format of data
- Determine size of data
- Determine computational capabilities
- Determine appropriate metrics

# Principles of scientific

- Exploration, Analysis, Presentation
- Determine format of data
- Determine size of data
- Determine computational capabilities
- Determine appropriate metrics
- Viewing the data and metrics

# Principles of scientific

- Exploration, Analysis, Presentation
- Determine format of data
- Determine size of data
- Determine computational capabilities
- Determine appropriate metrics
- Viewing the data and metrics
- Dissecting the data using the metrics

# Principles of scientific

- Exploration, Analysis, Presentation
- Determine format of data
- Determine size of data
- Determine computational capabilities
- Determine appropriate metrics
- Viewing the data and metrics
- Dissecting the data using the metrics
- **Annotating the data and metrics**

# General Design Principles

- Determining the correct quantity of information

# General Design Principles

- Determining the correct quantity of information
- Broader audiences require less complexity

# General Design Principles

- Determining the correct quantity of information to display: selecting an optimal maximum.
- Broader audiences require less complexity
- Specialized audiences can absorb greater complexity.

# General Design Principles

- Determining the correct quantity of information to display: selecting an optimal maximum.
- Broader audiences require less complexity
- Specialized audiences can absorb greater complexity.
- Seven elements, plus or minus two; per image.

# General Design Principles

- Determining the correct quantity of information to display: selecting an optimal maximum.
- Broader audiences require less complexity
- Specialized audiences can absorb greater complexity.
- Seven elements, plus or minus two; per image.
- Storytelling through animation allows for greater complexity to be simply presented

# Tufte's view of graphical excellence

- Show the data

# Tufte's view of graphical excellence

- Show the data
- Induce the viewer to observe substance

# Tufte's view of graphical excellence

- Show the data
- Induce the viewer to observe substance
- Avoid distortion

# Tufte's view of graphical excellence

- Show the data
- Induce the viewer to observe substance
- Avoid distortion
- Present many number in a small space

# Tufte's view of graphical excellence

- Show the data
- Induce the viewer to observe substance
- Avoid distortion
- Present many number in a small space
- Make large datasets coherent

# Tufte's view of graphical excellence

- Show the data
- Induce the viewer to observe substance
- Avoid distortion
- Present many number in a small space
- Make large datasets coherent
- Encourage viewer to make comparisons

# Tufte's view of graphical excellence

- Show the data
- Induce the viewer to observe substance
- Avoid distortion
- Present many number in a small space
- Make large datasets coherent
- Encourage viewer to make comparisons
- Reveal the data at several levels of detail

# Tufte's view of graphical excellence

- Show the data
- Induce the viewer to observe substance
- Avoid distortion
- Present many number in a small space
- Make large datasets coherent
- Encourage viewer to make comparisons
- Reveal the data at several levels of detail
- **Serve a clear purpose**

# Tufte's view of graphical excellence

- Show the data
- Induce the viewer to observe substance
- Avoid distortion
- Present many number in a small space
- Make large datasets coherent
- Encourage viewer to make comparisons
- Reveal the data at several levels of detail
- Serve a clear purpose
- Closely integrated with statistical & verbal descriptions

# Kellers' effective visualization

- Identify the visualization goal
- Remove mental road blocks
- Decide between data or phenomena

# Storytelling with animation

- Hollywood model

# Storytelling with animation

## Hollywood model

- Producer: defines the goal
- Director: manages the assembly
- Screenwriter: scripts the goal
- Production designer: creates the stage
- Cinematographer: visually captures the goal

# Storytelling with animation

- Hollywood model
- Defining the production goal

# Storytelling with animation

- Hollywood model
- Defining the production goal
- Managing the assembly

# Storytelling with animation

- Hollywood model
- Defining the production goal
- Managing the assembly
- Storyboards

# Storytelling with animation

- Hollywood model
- Defining the production goal
- Managing the assembly
- Storyboards
- Setting the stage

# Storytelling with animation

- Hollywood model
- Defining the production goal
- Managing the assembly
- Storyboards
- Setting the stage
- Directing the eye of the camera

# Storytelling with animation

- Hollywood model
- Defining the production goal
- Managing the assembly
- Storyboards
- Setting the stage
- Directing the eye of the camera
- Reflection & re-invention

# Living within a budget

- Defining your requirements

# Living within a budget

## Defining your requirements

- Storage media
- Disk space
- Memory
- General computation
- Graphics computation
- Display
- Commercial off the shelf software
- Custom software

# Living within a budget

- Defining your requirements
- NCMI: Computers, graphics & OpenGL

# Living within a budget

- Defining your requirements
- NCMI: Computers, graphics & openGL
- NCMI: Iris Explorer scientific visualization s/w

# Living within a budget

- Defining your requirements
- NCMI: Computers, graphics & openGL
- NCMI: Iris Explorer scientific visualization s/w
- NCMI: Amira scientific visualization s/w

# Living within a budget

- Defining your requirements
- NCMI: Computers, graphics & openGL
- NCMI: Iris Explorer scientific visualization s/w
- NCMI: Amira scientific visualization s/w
- NCMI: Scientific Animation & Integration Library

# Basic viz tools

- Reading the data

# Basic viz tools

- Reading the data
- Creating geometry

# Basic viz tools

- Reading the data
- Creating geometry
- Coloring of geometry

# Basic viz tools

- Reading the data
- Creating geometry
- Coloring of geometry
- Creating transformations

# Basic viz tools

- Reading the data
- Creating geometry
- Coloring of geometry
- Creating transformations
- Manipulating the camera

# Basic viz tools

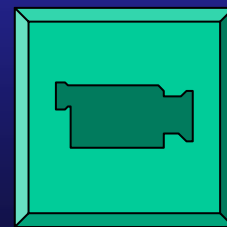
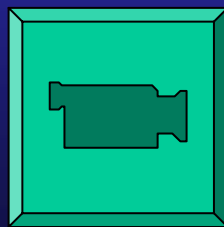
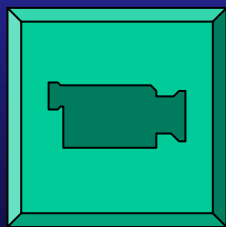
- Reading the data
- Creating geometry
- Coloring of geometry
- Creating transformations
- Manipulating the camera
- Rendering the image

# Examples

- Electron microscopy
- X-ray crystallography

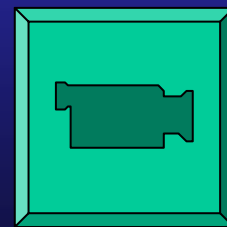
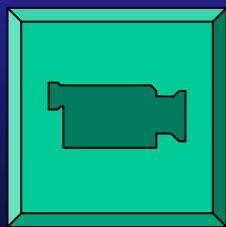
# Electron microscopy

- Collaborator: H. Zhou/UT Houston
- Subject: Rice Dwarf Virus@8Å
- Comment: examination of helices and protein interactions. Creating several levels of detail.



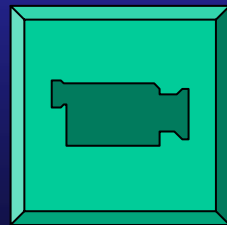
# X-ray crystallography

- Subject: BTV & actin protein folding
- Comment: examination of sequence using PDB data. Generation of simulated electron density.



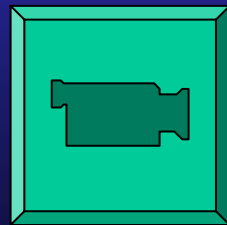
# Electron microscopy

- Collaborator: I. Serysheva/BCM
- Subject: Calcium Release Channel
- Comment: modeling of functional dynamics.



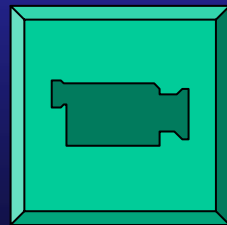
# Electron microscopy

- Collaborator: M. Sherman/Purdue
- Subject: LDL
- Comment: use of an ellipsoid metric to dissect the data.



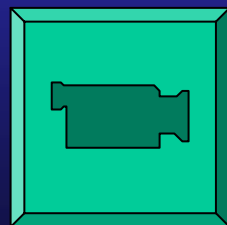
# X-ray crystallography

- Collaborator: F. Quioco/BCM
- Subject: ADSS binding sites
- Comment: examination of conformational changes.



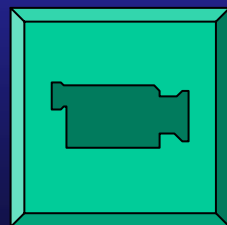
# Electron microscopy

- Collaborator: E. Orlova/Birbeck College
- Subject: SPP1
- Comment: comparison of homologous structures.



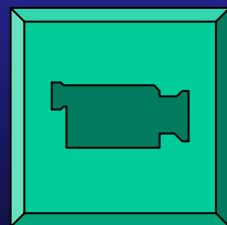
# Electron microscopy

- Collaborator: W. Chiu/BCM
- Subject: HSV
- Comment: examination of DNA core using a geometric model.



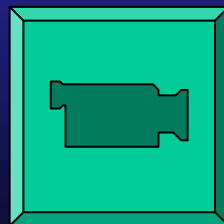
# Electron microscopy

- Collaborator: W. Chiu/BCM
- Subject: HSV
- Comment: dissection of triplex into constituent proteins. Examination of slices.



# EM & X-ray

- Collaborator: M. Baker & B.
- Subject: examination of HSV VP5
- Comment: portions determined by x-ray are compared to EM data.  
Electrostatic data is applied.



# *Suggested reading*

- E. Tufte, ***Visual Explanations***, Graphics Press, 1997
- P. & M. Keller, ***Visual Cues***, IEEE Press, 1992
- J. Vince, ***3D Computer Animation***, Addison-Wesley, 1992
  
- E. Tufte, ***The Visual Display of Quantitative Information***, Graphics Press, 1983
- Issac Kerlow, ***The Art of 3D Computer Animation and Imaging***, Van Norstrand Reinhold, 1996
- S. Katz, ***Film Directing Shot by Shot***, Michael Wiese Productions, 1991
  
- ***For further information contact me at matthewd@BCM.TMC.EDU***