数据驱动的声明式可视化动画的研究与构建

Declarative Data-Driven Chart Animation

葛彤

目录

TOC

目录

TOC

动画

动画为人们对静态事物的深层理解提供了一种有效的手段。

描述变化规律

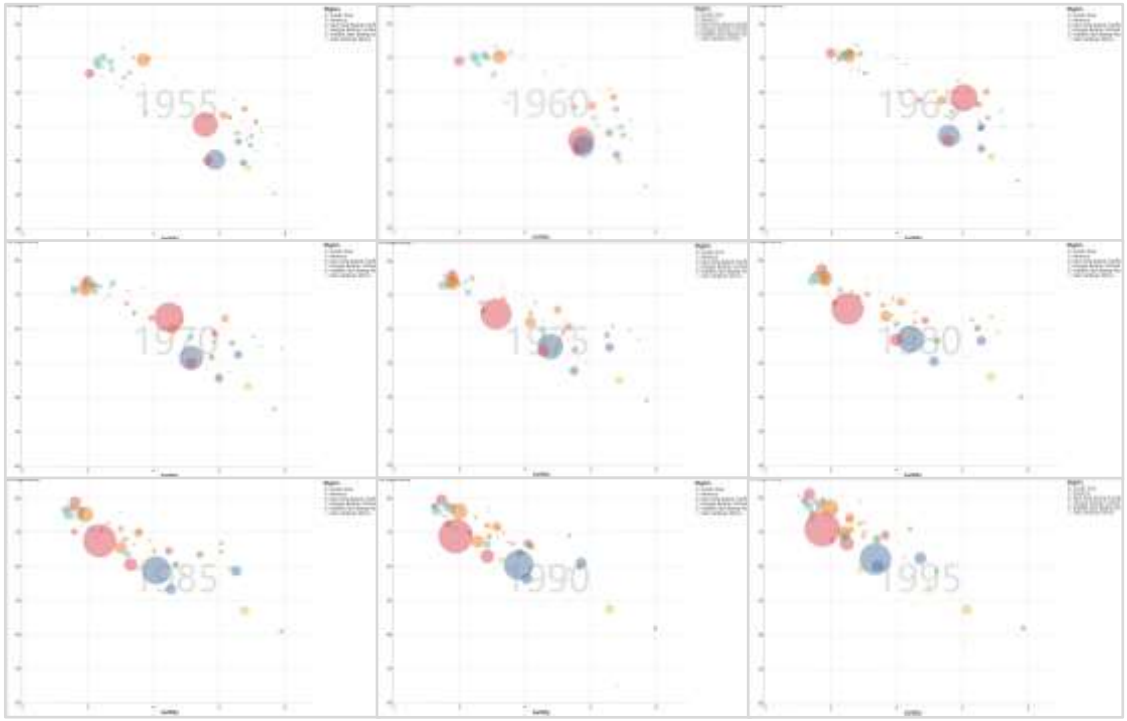
展示状态切换

展开叙事过程

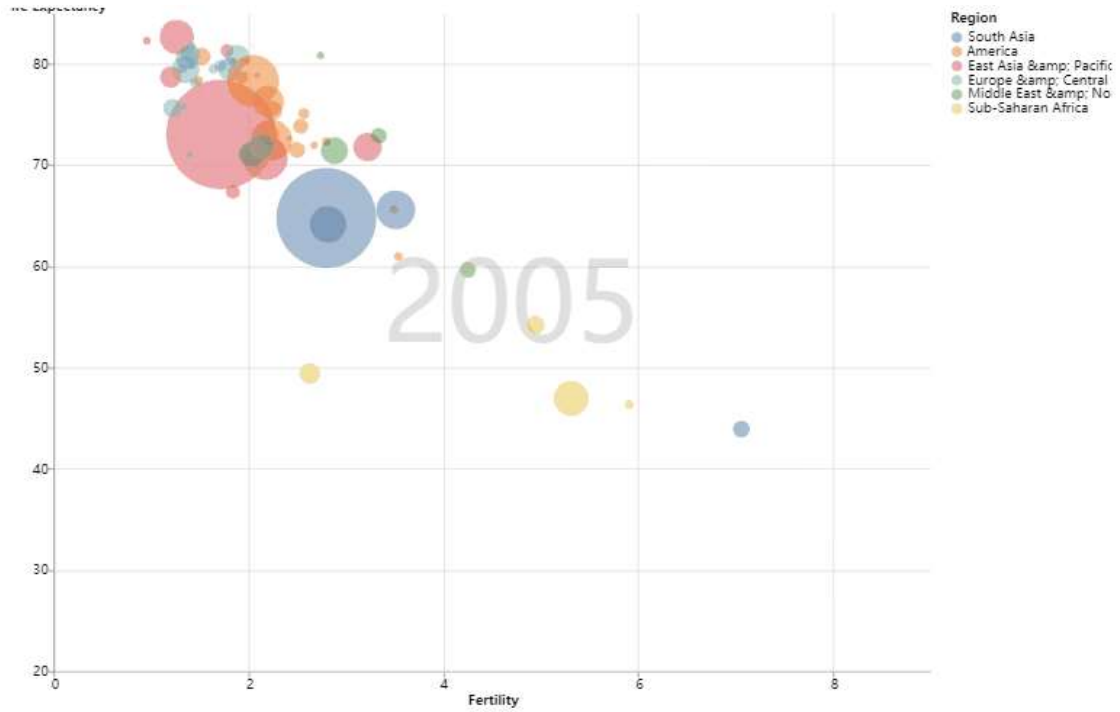
建立联系　认知恒常　吸引注意　表达情绪

动画

图表动画



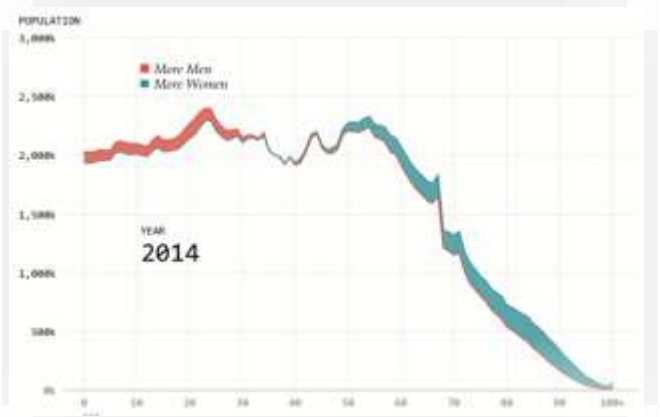
静态图表



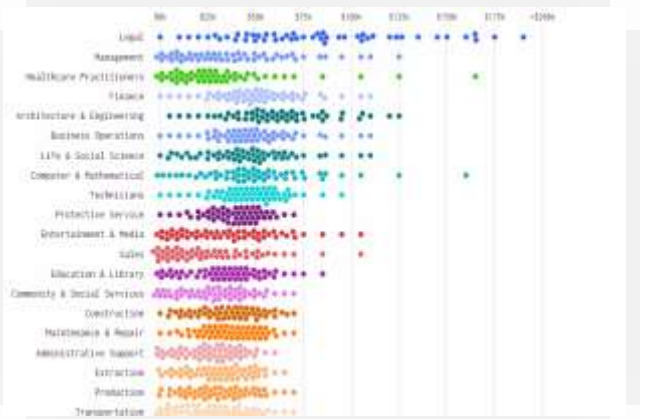
图表动画

图表动画的广泛应用

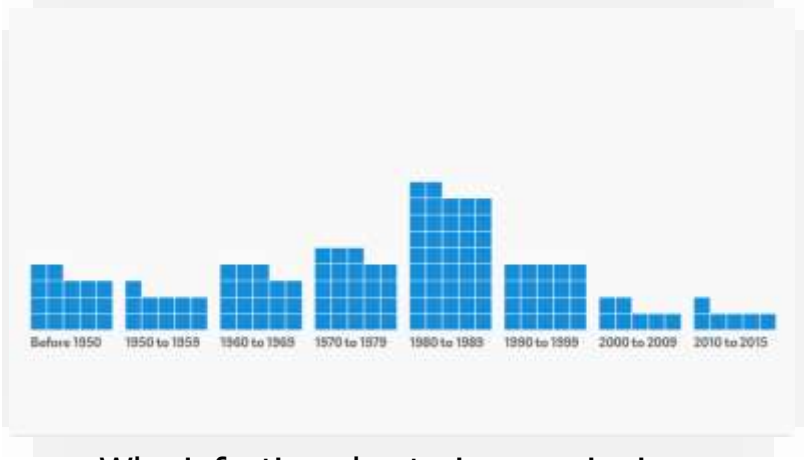
Punishing Reach of Racism for Black Boys



Population in the United States

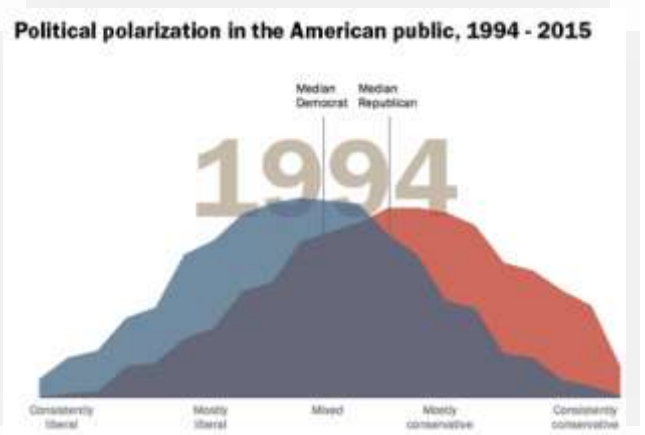


Shifting Incomes for American Jobs



Why infectious bacteria are winning

Out of sight, out of mind



Political polarization in the American public

研究目标

问 题

缺乏针对可视化动画的高效直观的构建方法

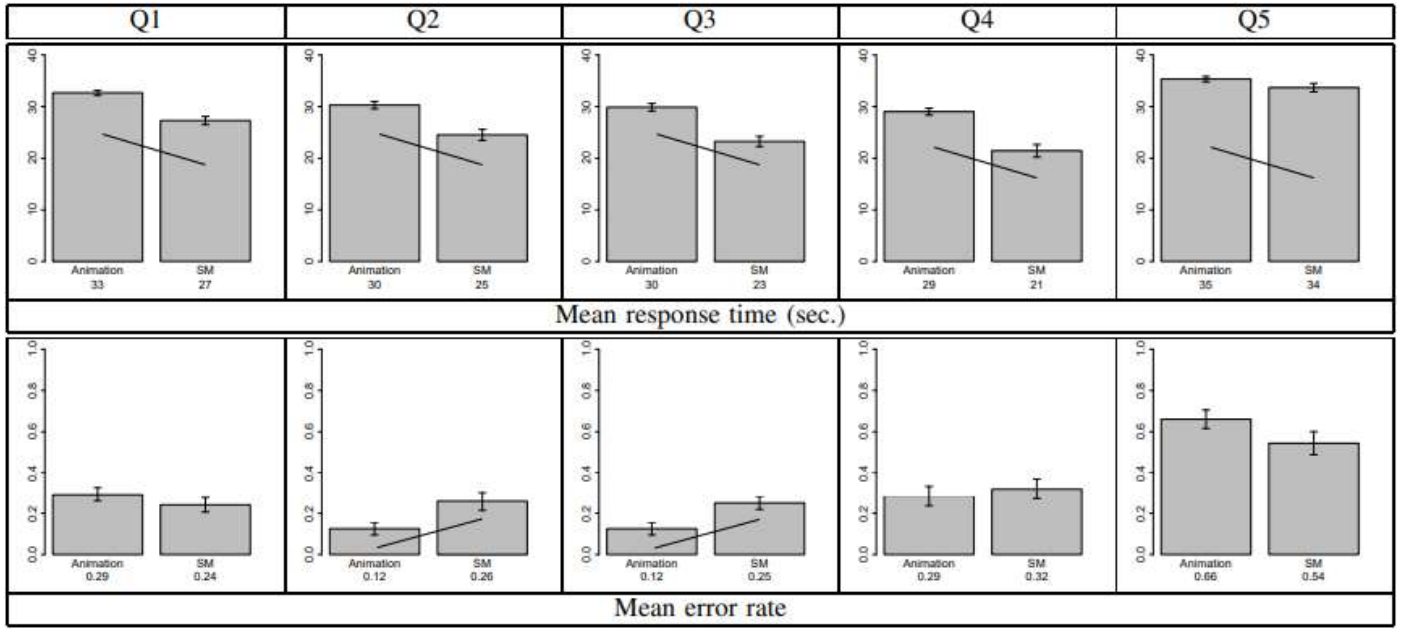
简洁精确的描述方式

数据驱动的构建过程

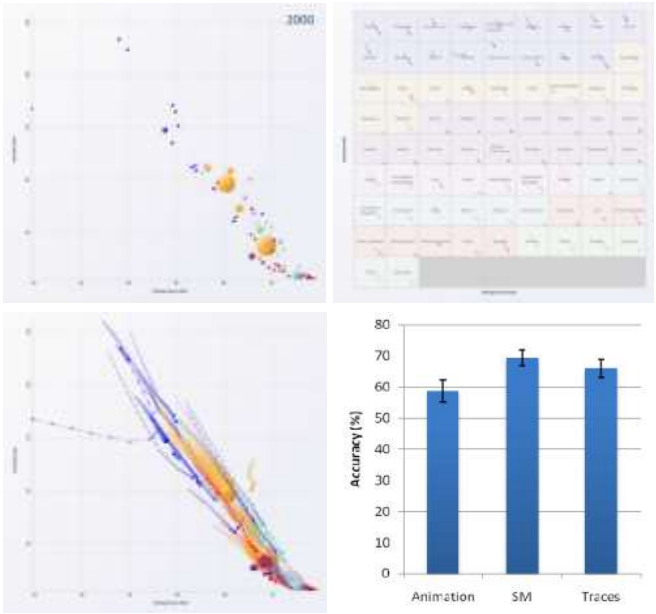
直观自然的交互模式

数据驱动可视化动画的直观自然的创作模式

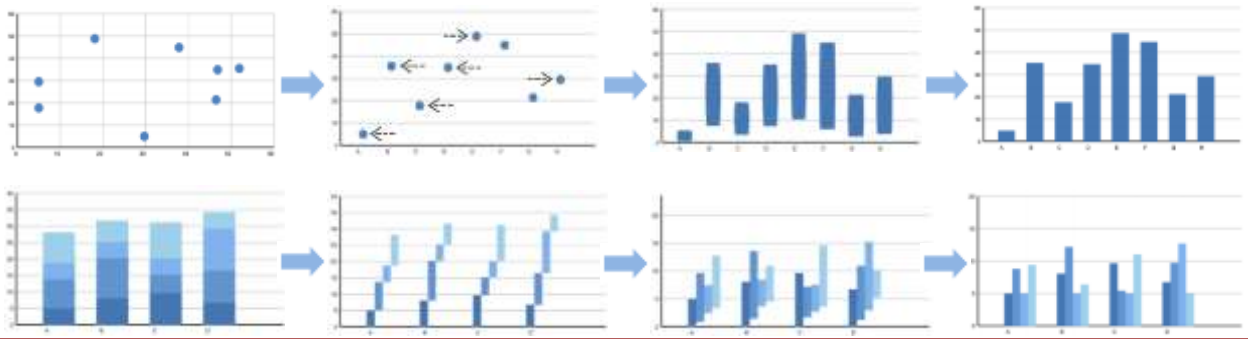
图表动画的有效性研究



[Archambault et al. 2010]
动画更容易让读者跟踪状态的改变

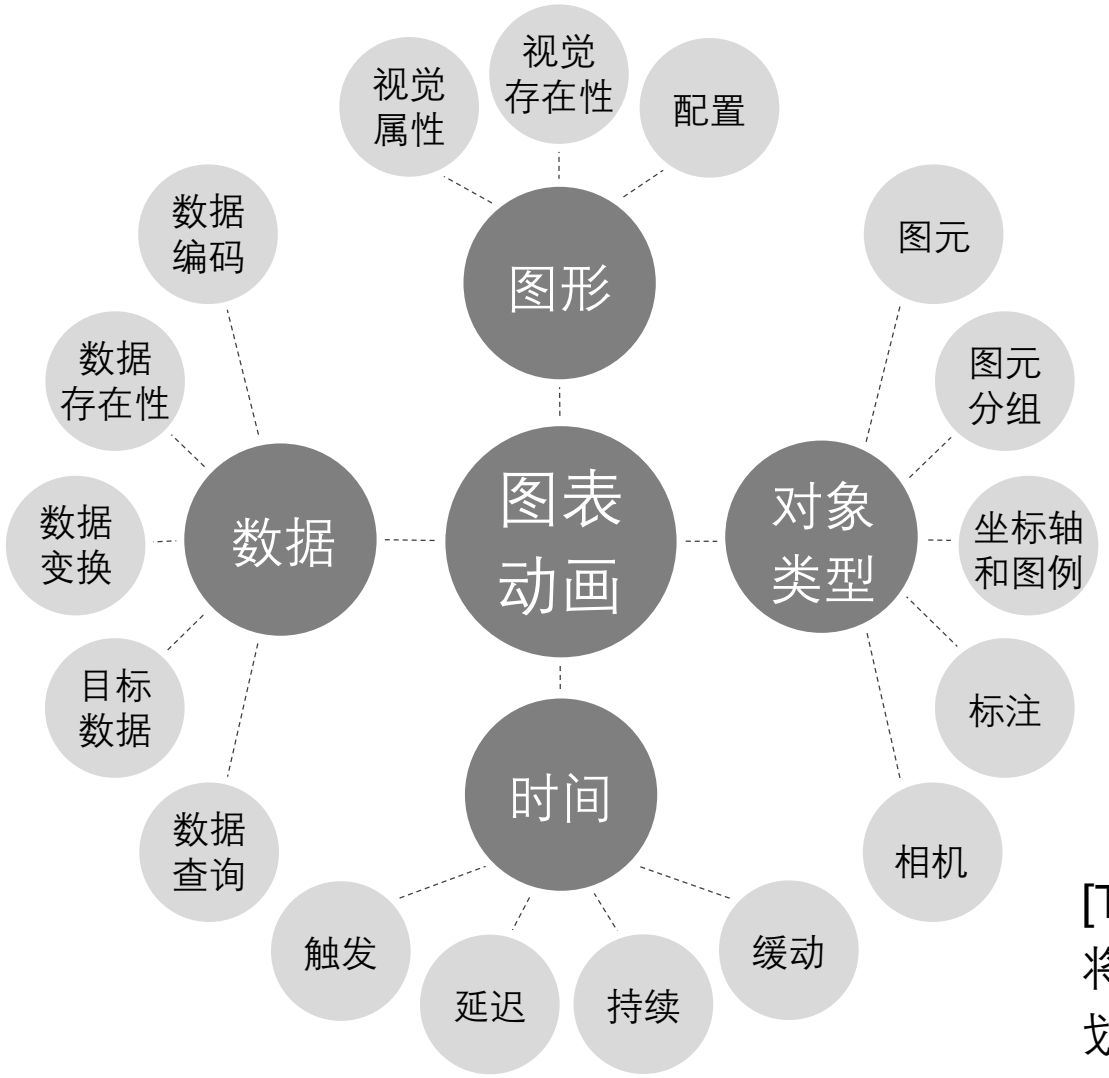


[Brehmer et al. 2008, Robertson et al. 2008]
带有动画效果的散点图在趋势可视化中更为有效



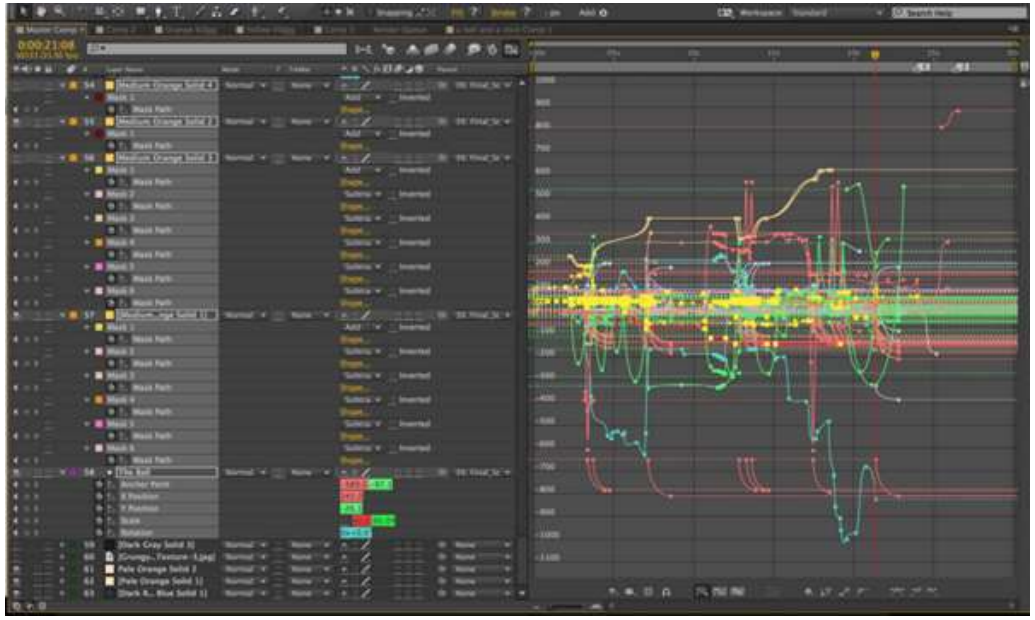
[Heer and Robertson, 2010]
为了更有效的跟踪状态变化提出的一系列过渡动画类型

图表动画的设计空间



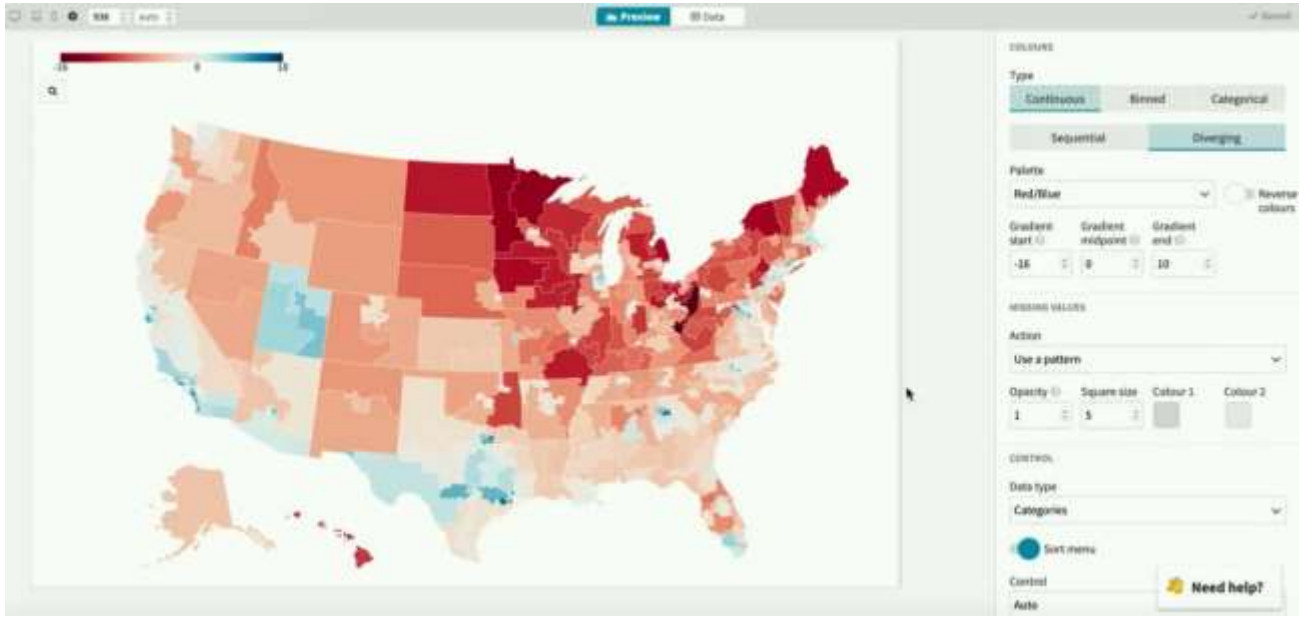
[Thompson et al. 2020]
将图表动画设计空间
划分为4个维度

图表动画的生成（交互式系统）



基于关键帧的工具（After Effects等）：

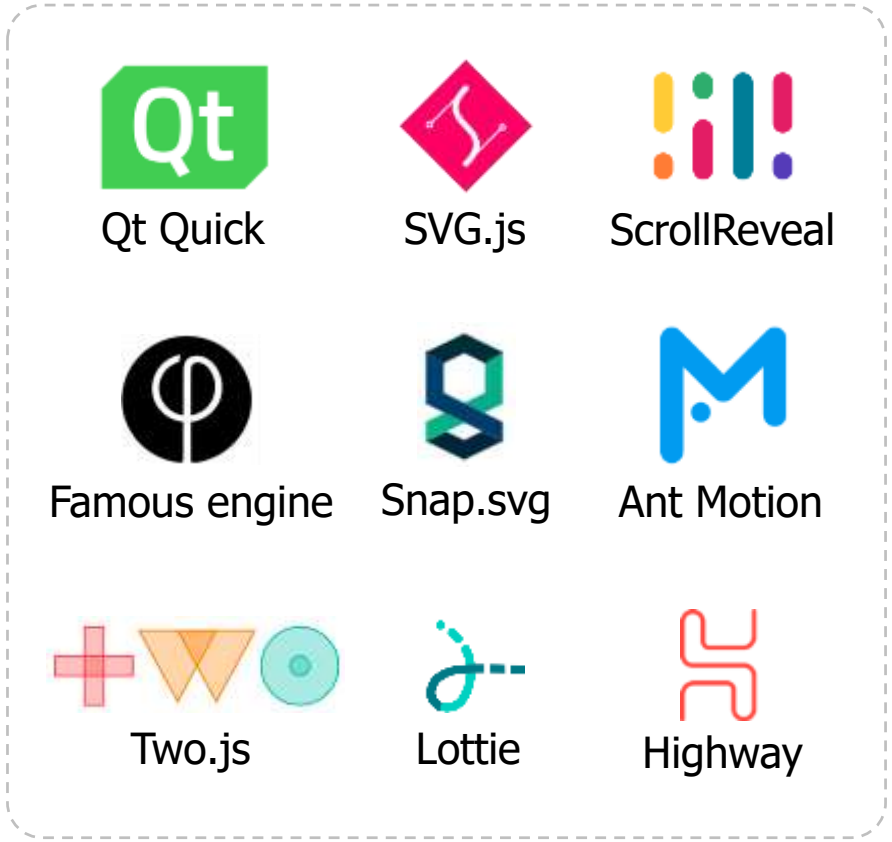
- 支持复杂生动的动画效果
- 操作繁琐、准确率没有保障



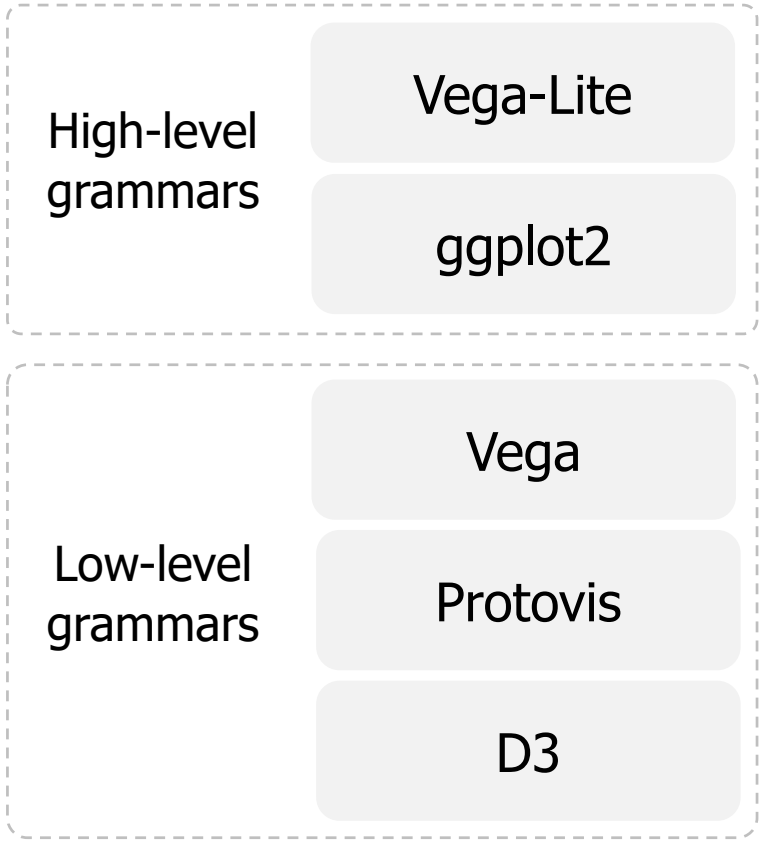
基于模板的工具（Flourish等）：

- 方便使用
- 不支持定制化的图表动画

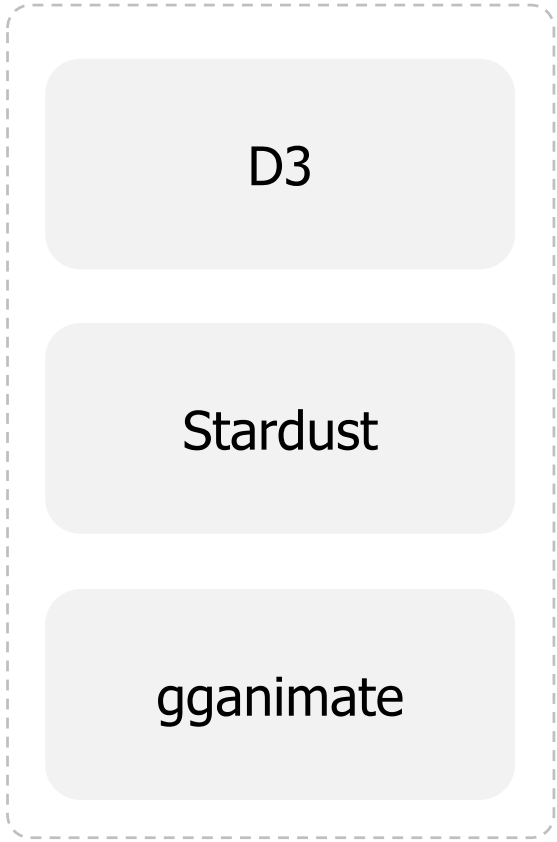
图表动画的生成（可视化及动画库）



图形界面动画库



可视化编程语言



可用于动画创作的可视化编程语言

存在问题

1

缺乏对**数据**、**视觉编码**、**时序**三者关系的抽象，导致**动画表达与数据感知的一致性得不到保障**。

2

缺乏对动画的**直观表达**形式和**自动化构建**方法，导致动画过程**难以想象和理解**，**交互繁琐复杂**。

3

缺乏**直接的交互形式**，导致用户的**动画表达**与**交互实现**之间存在认知处理，增加了学习成本。

解决方案

支持数据驱动，简洁精确的可视化动画描述方式

Canis：数据驱动的可视化动画编程语言

基于动画设计空间（目标、划分、时序、效果），数据驱动的声明式可视化动画编程语法 。

解决方案

支持数据驱动，简洁精确的可视化动画描述方式

- 动画的表现形式
- 数据驱动的动画构建过程优化

CAST (CAnis STudio)：可视化动画交互式创作平台

Canis：数据驱动的可视化动画编程语言

语法扩展

编译优化

融合了用于表述动画过程的视觉规范和数据驱动的动画自动补全的交互式创作平台。

目录

TOC

研究内容

CAST (CAnis STudio)：可视化动画交互式创作平台

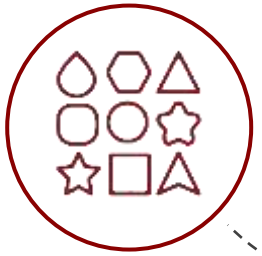
Canis：数据驱动的可视化动画编程语言

- 语法扩展
- 编译优化

Canis – 目标

有效性

可轻松建立数据、视觉编码、时序三者之间的关系，以数据驱动的方式构建可视化动画



表现力

支持为各类图表指定丰富的动画过程

易用性

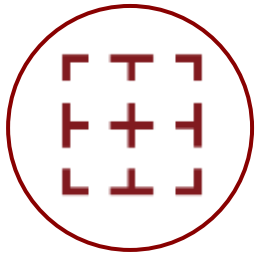
语法精确简洁，编译输出的动画体积小且拥有较高渲染性能

平衡

Canis – 设计原则

DP1: 高级语法规范

解耦编码与实现逻辑，并使用简洁易懂的高级语法规范。



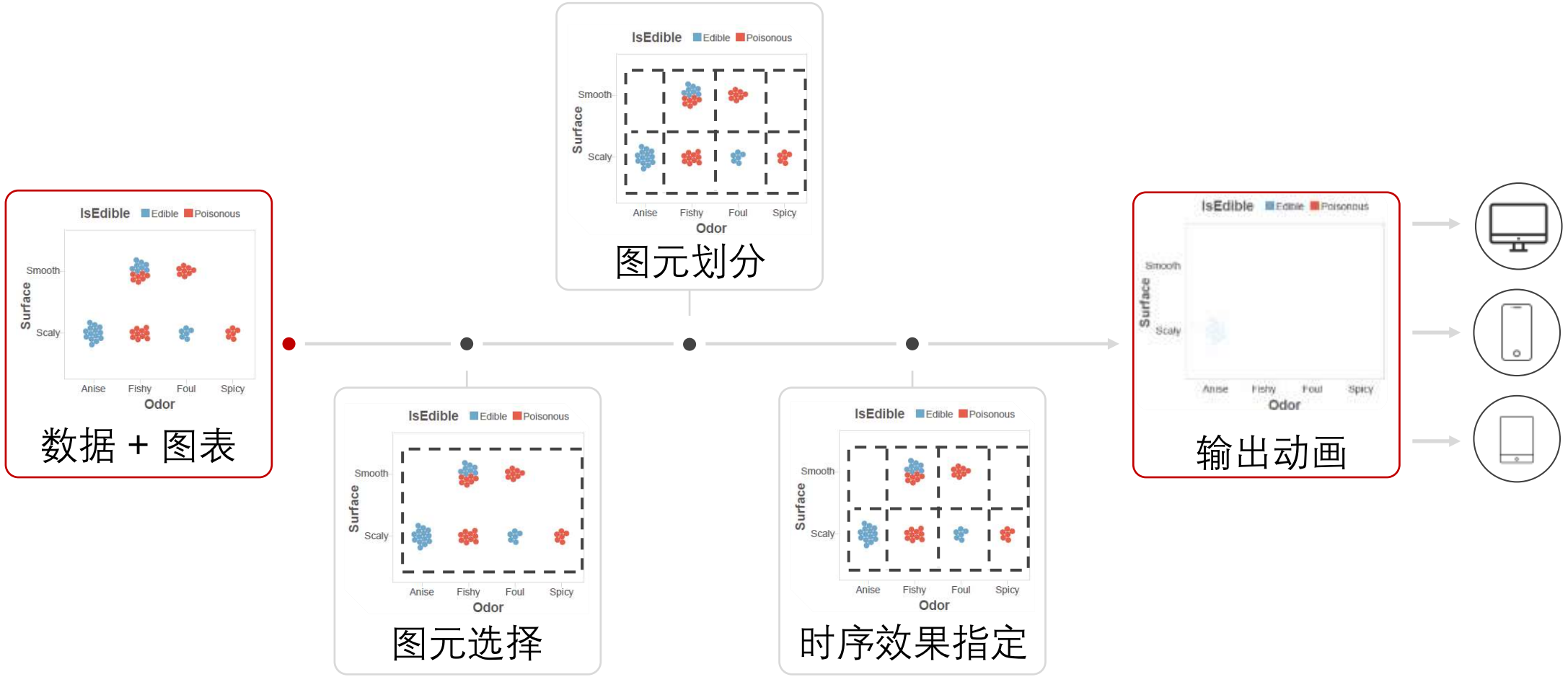
DP2: 有意义的划分和时序

借助数据属性高效的对图元进行分组管理和动画时序定义。

DP3: 支持跨平台应用

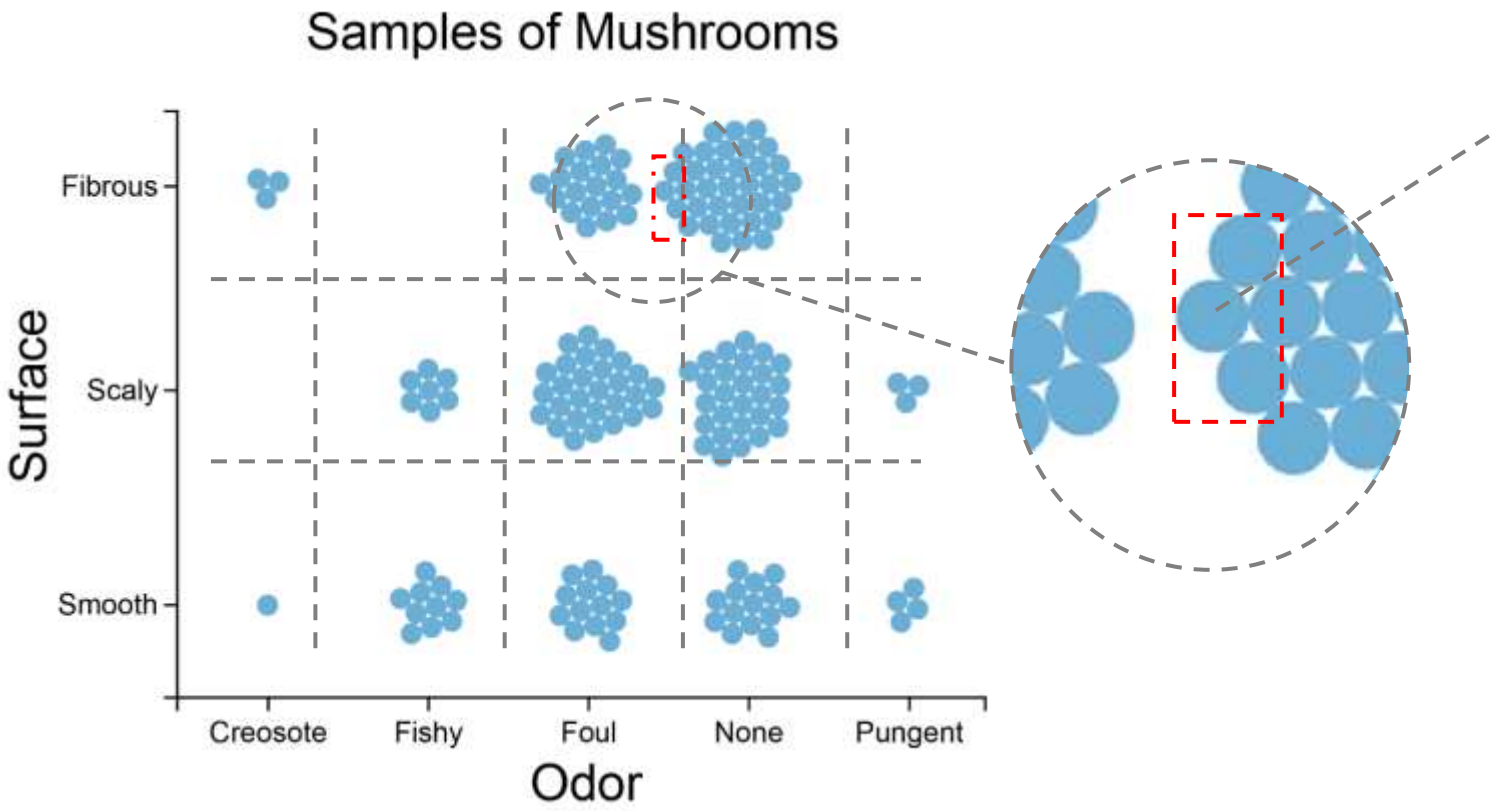
支持跨平台应用以提高适用性和实用性。

Canis – 结构


IsEdible Edible Poisonous
Smooth
Scaly
Surface
Anise Fishy Foul Spicy
Odor
数据 + 图表
图元划分
图元选择
时序效果指定
输出动画

Canis – 输入（dSVG）

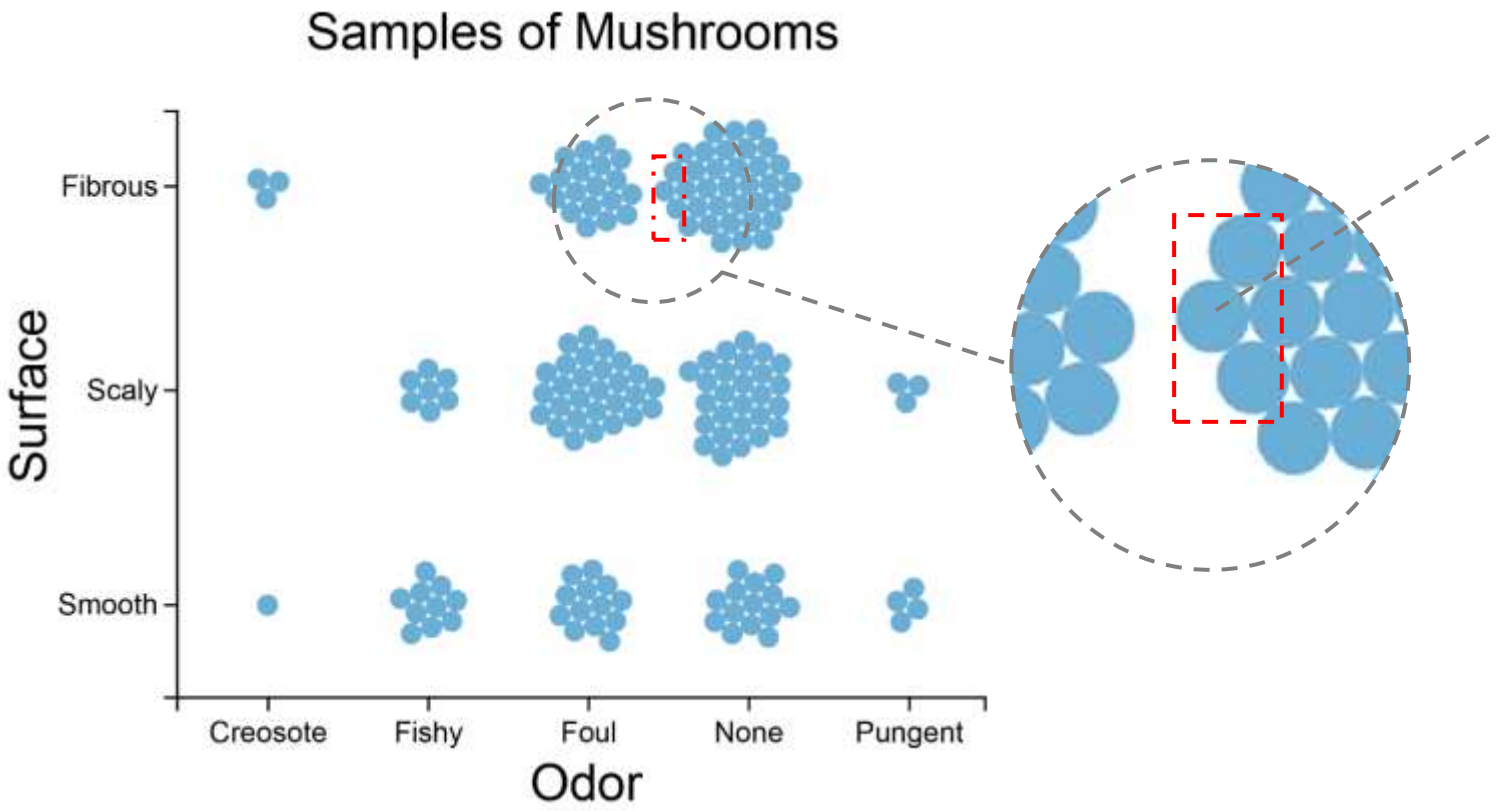
SVG图表中所能够提供的视觉信息不足以分析出图元的数据关系。




```
<circle
    cx="709"
    cy="86"
    r="4.5"
    fill="#6baed6"
></circle>
```

Canis – 输入（dSVG）

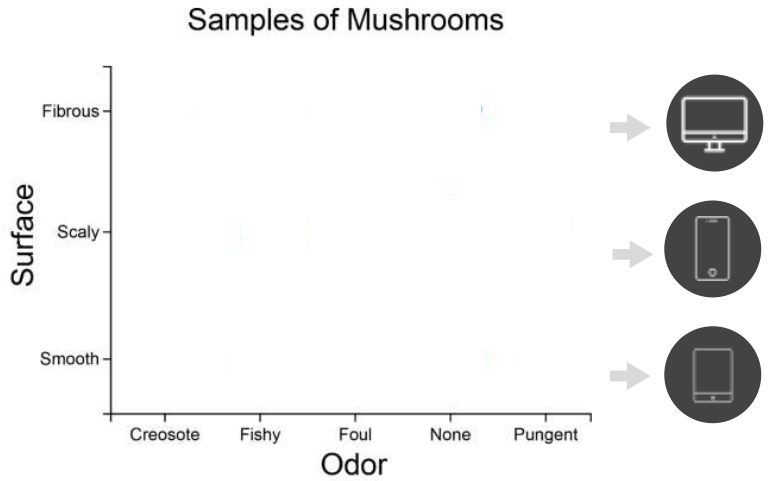
为了支持数据驱动的图元划分和时序定义（DP2），Canis以内嵌数据的SVG图表（dSVG）作为输入：




```
<circle
    cx="709"
    cy="86"
    r="4.5"
    fill="#6baed6"
    id="mark109"
    class="mark circle"
    datum="{
      'Odor':'None',
      'Surface':'Fibrous'
    }"
></circle>
```

Canis – 输出（Lottie）

为了支持跨平台应用（DP3），Canis会将用户指定的动画编译为Lottie编码。

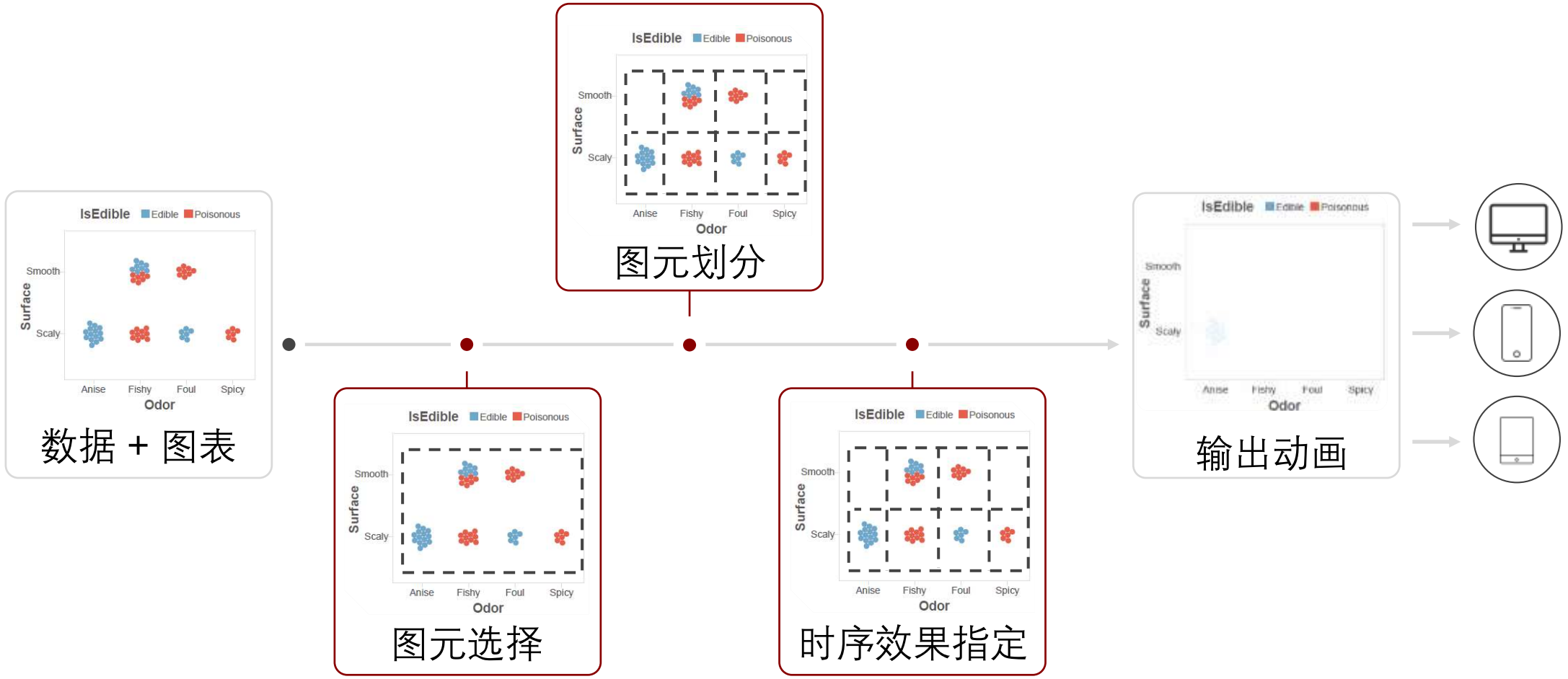



```
{
  "ty": 4,"ddd": 0,"sr": 1,
  "ks": {
    "o": { "a": 0, "k": 100 },
    "p": { "a": 0, "k": [0, 0, 0]},
    "a": { "a": 0, "k": [0, 0, 0]},
    "s": { "a": 0, "k": [100, 100, 100]}},
  "shapes": [...],
  "completed": true
}
```

WHY

- 支持多平台本地渲染
- 动画体积更小，渲染效率更高

Canis – 结构


IsEdible Edible Poisonous
Smooth
Scaly
Surface
Anise Fishy Foul Spicy
Odor
数据 + 图表
图元划分
图元选择
时序效果指定
输出动画

Canis – 语法

Canis通过一系列**动画单元**（aniunit）对输入**图表**（chart）的动画进行描述，以及**分视图操作符**（facet）帮助生成多视图动画。

```
{
  "charts": [],
  "facet": [],
  "aniunits": []
}
```

Canis – 语法

Canis通过一系列**动画单元**（aniunit）对输入**图表**（chart）的动画进行描述，以及**分视图操作符**（facet）帮助生成多视图动画。

```
{
  "charts": [],
  "facet": [],
  "aniunits": []
}
```

输入dSVG图表

单一图表

```
{"source": "./charts/mushrooms.dsvg"}
```

多图表

```
{
  "source": "./charts/polio_s1.dsvg"
},{
  "source": "./charts/polio_s2.dsvg"
}
```

或

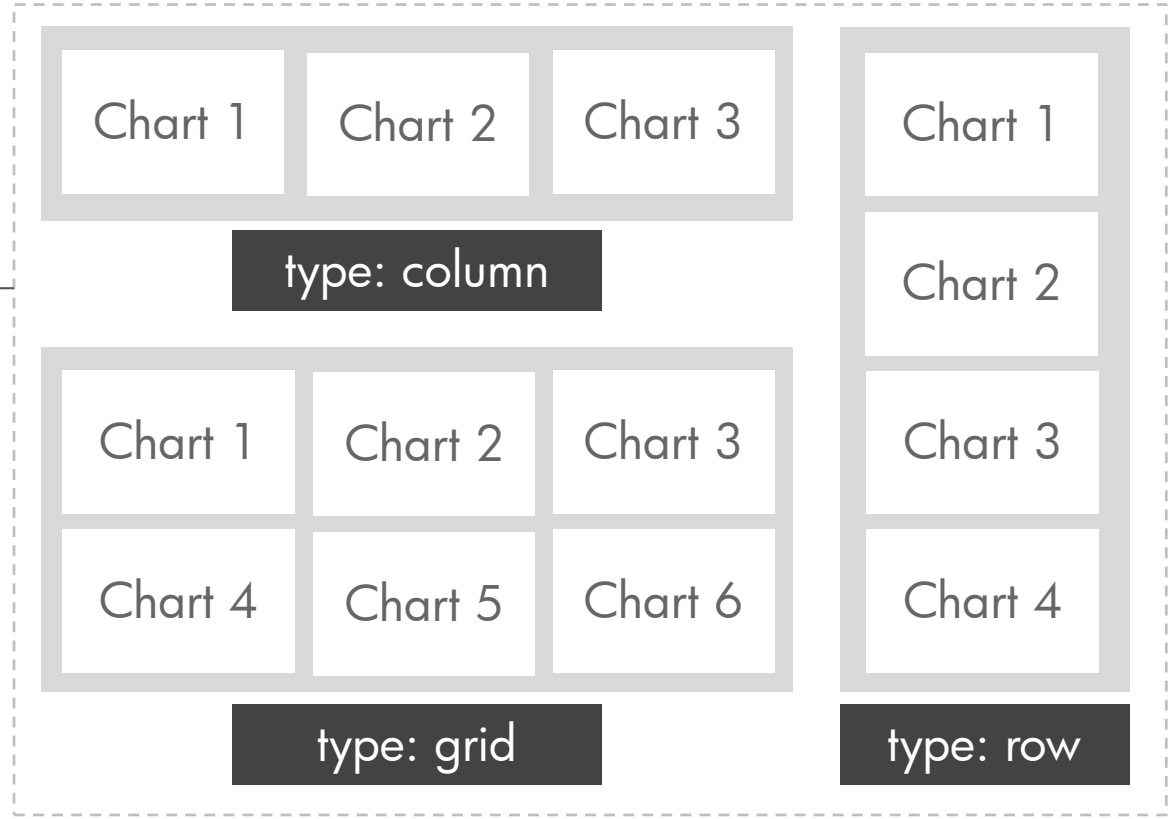
```
{
  "id": "polio",
  "source": "./charts/gapminder/polio_s",
  "start": 1,
  "end": 2
}
```

用于分视图操作符

Canis – 语法

Canis通过一系列**动画单元**（aniunit）对输入**图表**（chart）的动画进行描述，以及**分视图操作符**（facet）帮助生成多视图动画。

```
{
  "charts": [],
  "facet": [{
      "type": ...,
      "views": []
  }],
  "aniunits": []
}
```



Canis – 语法

Canis通过一系列**动画单元**（aniunit）对输入**图表**（chart）的动画进行描述，以及**分视图操作符**（facet）帮助生成多视图动画。

```
{
  "charts": [],
  "facet": [{
      "type": ...,
      "views": []
  }],
  "aniunits": []
}
```

```
"charts": [{
    "id": "africa",
    "source": "...",
    "start": 0,
    "end": 9
  },{
    "id": "america",
    "source": "...",
    "start": 0,
    "end": 9
}]
```

```
"facet": {
    "type": "column",
    "views": [{
        "range": [
          "africa0",
          "africa9"
        ]},{
        "range": [
          "america0",
          "america9"
        ]}
    ]}
}
```

两个动画视图并排布局

Canis – 语法

Canis通过一系列**动画单元**（aniunit）对输入**图表**（chart）的动画进行描述，以及**分视图操作符**（facet）帮助生成多视图动画。

```
{
  "charts": [],
  "facet": [],
  "aniunits": [{
    "selector": ...,
    "grouping": ...,
    "reference": ...,
    "delay": ...,
    "effects": []
  }
]}
```

What：指定动画单元中的动画内容

基于数据对图元进行分组

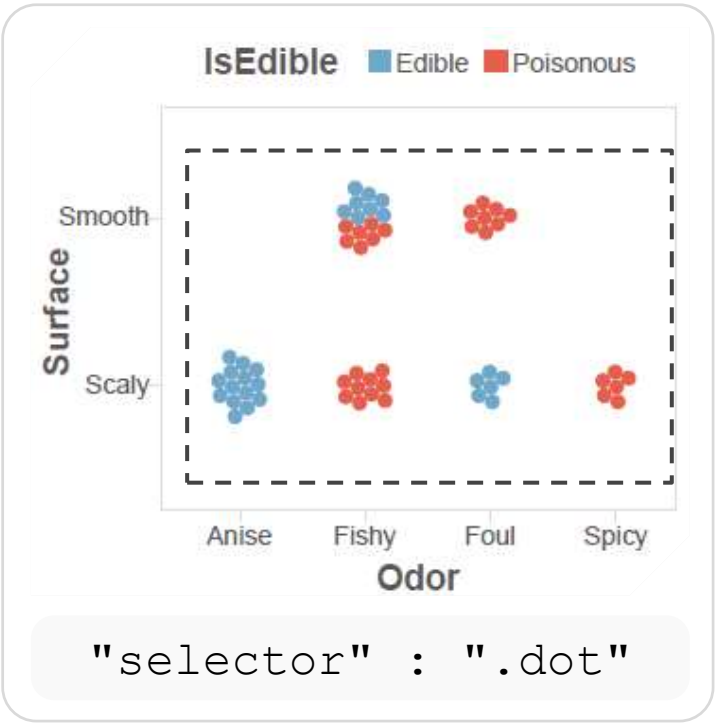
When：决定了动画单元的开始时间

How：绑定动画效果

Canis – 语法

选择器（selector）使用W3C选择器API来选择动画单元中所涉及到的图元。

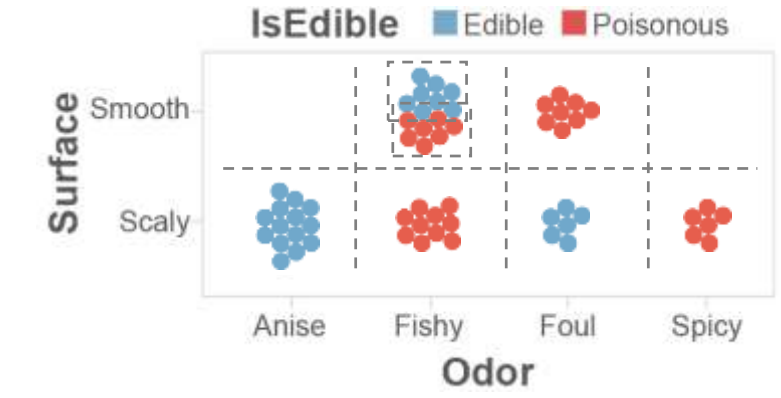
```
{
  ...,
  "aniunits": [{
    "selector": ...,
    "grouping": ...,
    "reference": ...,
    "delay": ...,
    "effects": []
  }]
}
```

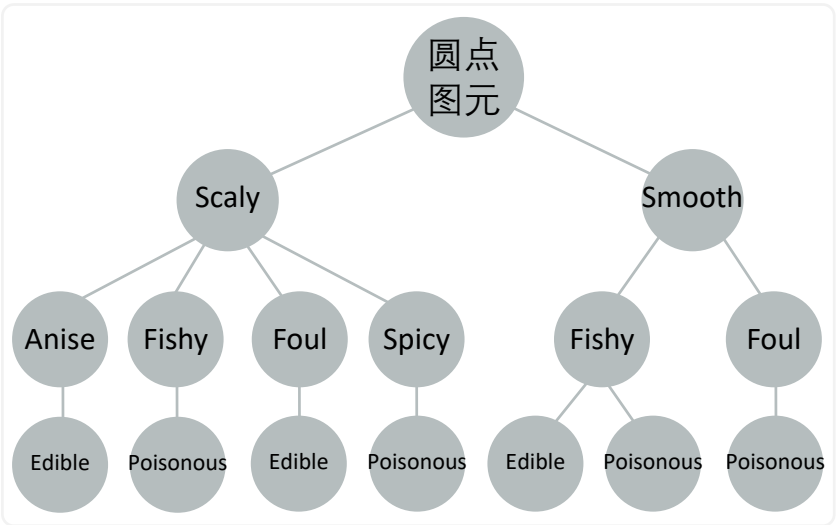


Canis – 语法

划分（grouping）为动画单元的核心属性，它将图元按照数据属性进行层次化分组，并依据各级分组定义图元动画的时序关系。

```
{
  ...,
  "aniunits": [{
    "selector": ...,
    "grouping": {
      "groupBy": ...,
      "sort": ... ,
      "reference": ...,
      "delay": ...,
      "grouping": ...
    },
    ...
  }]
}
```



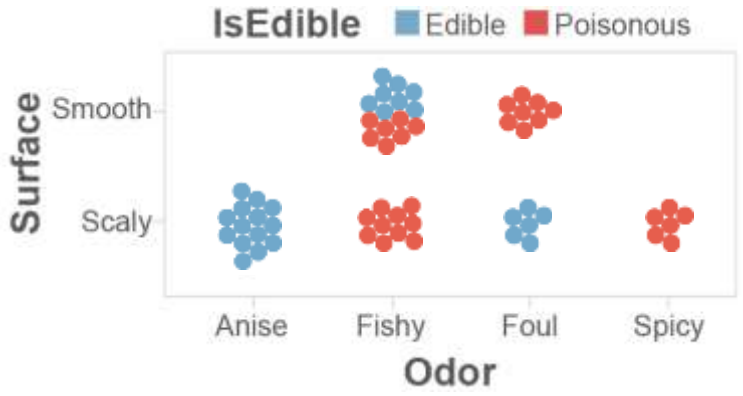



```
"selector": ".dot",
"grouping": {
  "groupBy": "Surface",
  "reference":
     "start after previous",
  "grouping": {
    "groupBy": "Odor",
    "reference":
       "start after previous",
    "grouping": {
      "groupBy": "IsEdible",
      "reference":
         "start after previous",
}}},
```

Canis – 语法

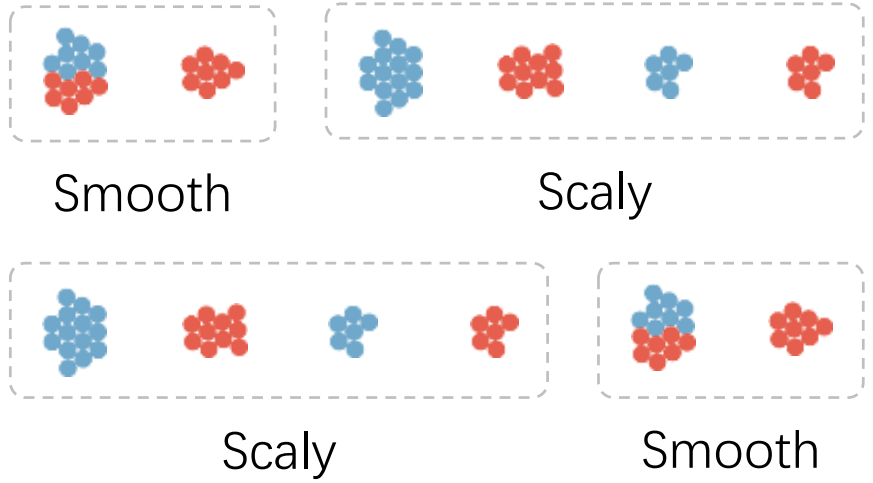
划分之后的图元分组可进一步根据数据值指定动画顺序。

```
{
  ...,
  "aniunits": [{
    "selector": ...,
    "grouping": {
      "groupBy": ...,
      "sort": ... ,
      "reference": ...,
      "delay": ...,
      "grouping": ...
    },
    ...
  }]
}
```




```
"sort": {
  "field": "Surface",
  "order": ["Smooth", "Scaly"]
}
```

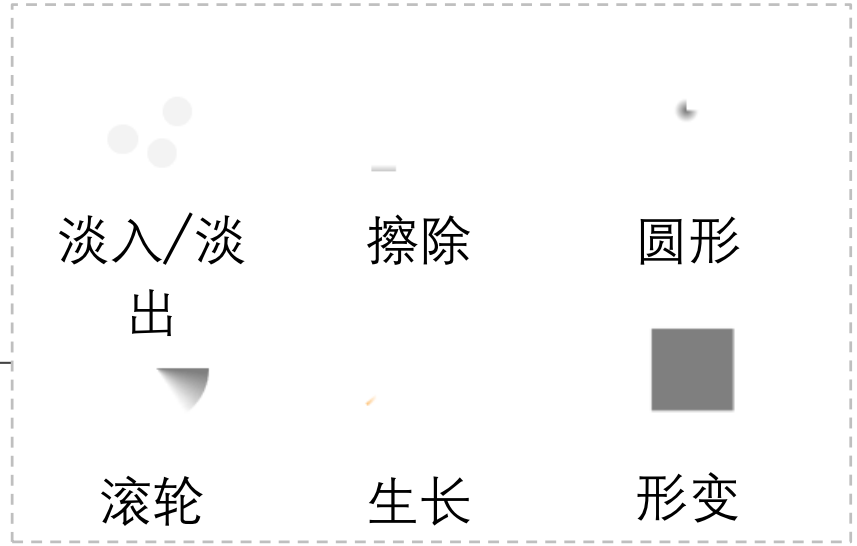
```
"sort": {
  "field": "Surface",
  "order": ["Scaly", "Smooth"]
}
```



Canis – 语法

每个动画图元可以绑定若干个**动画效果**（effect）。

```
{
  ...,
  "aniunits": [{
    ...,
    "effects": [{
        "reference": ...,
        "delay": ...,
        "type": ...,
        "easing": ...,
        "Duration": ...
    }]
  }
]}
```



Canis – 编译机制

Parse

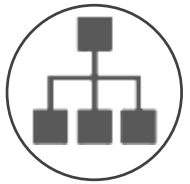
纠错、补充缺省值，构建完整动画声明

Bind

使用视觉属性描述动画效果，并绑定至图元单元

Translate

转译为Lottie编码



Build

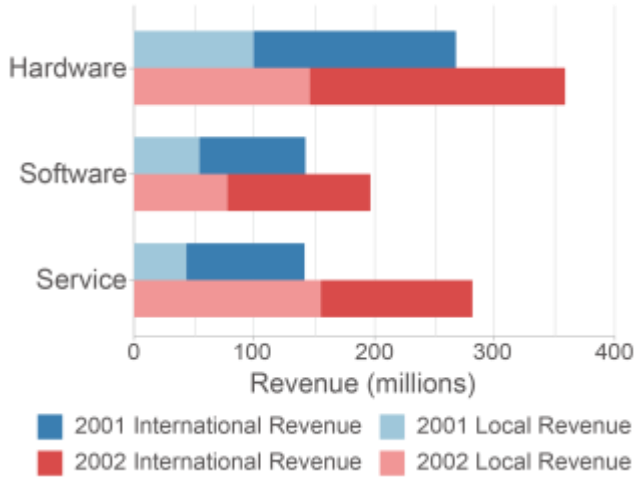
划分图元，生成图元单元的层次结构

Evaluate

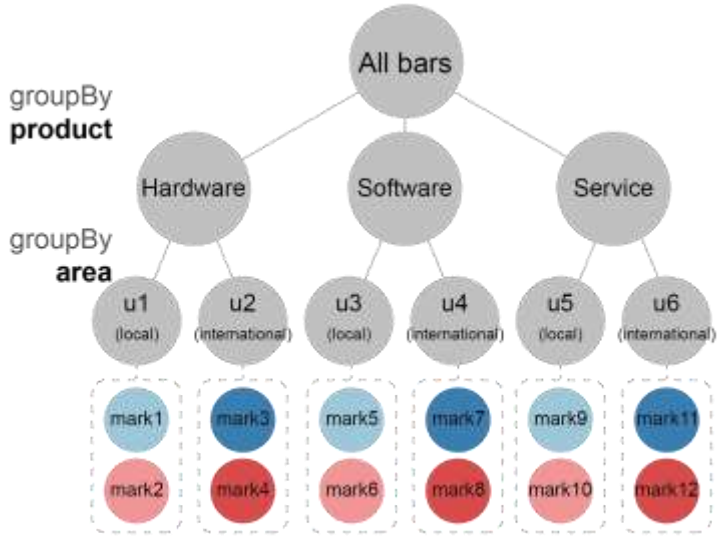
计算图元单元的动画时间

Canis – 编译机制（Build）

Build阶段根据数据构建图元单元层次树，并更新图元单元表（mark unit table）和关键帧表（keyframe table）



输入图表



图元单元层次树

keyframe table

unit_id	effect_id	duration(ms)	start_time(s)
u1			
u2			
u3			
u4			
u5			
u6			

mark unit table

unit_id	marks
u1	mark1, mark2
u2	mark3, mark4
u3	mark5, mark6
u4	mark7, mark8
u5	mark9, mark10
u6	mark11, mark12

Canis – 编译机制（Bind）

Bind阶段使用底层视觉属性实现指定动画效果并绑定到图元单元，同时更新动画效果表（animation effect table），视觉属性表（mark channel table）和关键帧表（keyframe table）。

视觉属性实现
动画效果

蒙版辅助实现动画效果

keyframe table

unit_id	effect_id	duration (ms)	start_time(s)
u1	e1	300	
u2	e1	300	
u3	e1	300	
u4	e1	300	
u5	e1	300	
u6	e1	300	

animation effect table

effect_id	channel	from	to	easing
e1	width	0.00	1.00	linear

mark channel table

mark_id	width
mark1	100
mark2	147
mark3	167
⋮	⋮
mark10	156
mark11	97
mark12	125

Canis – 编译机制（Evaluate）

Evaluate阶段根据编码中各级时序关系定义，计算各图元动画开始时间，并更新关键帧表 （keyframe table）。

keyframe table

unit_id	effect_id	duration(ms)	start_time(s)
u1	e1	300	0
u2	e1	300	0.4
u3	e1	300	0.9
u4	e1	300	1.3
u5	e1	300	1.8
u6	e1	300	2.2

mark unit table

unit_id	marks
u1	mark1, mark2
u2	mark3, mark4
u3	mark5, mark6
u4	mark7, mark8
u5	mark9, mark10
u6	mark11, mark12

animation effect table

effect_id	channel	from	to	easing
e1	width	0.00	1.00	linear

mark channel table

mark_id	width
mark1	100
mark2	147
mark3	167
⋮	⋮
mark10	156
mark11	97
mark12	125

Canis – 编译机制（Translate）

通过模板复用机制减少冗余声明，模板分为两类：

- 静态模板：字体、图片、形状
- 动态模板：视觉属性随着时间的变化方式

```
"shape": {
  "type": "rectangle",
  "width": 60,
  "height": 20,
  "fill": "#5b9bd5"
},
"effect": [{
  "time": 0,
  "opacity": 0
},{
  "time": 0.3,
  "opacity": 1
}]
```

```
"shape": {
  "type": "rectangle",
  "width": 90,
  "height": 20,
  "fill": "#5b9bd5"
},
"effect": [{
  "time": 0.3,
  "opacity": 0
},{
  "time": 0.6,
  "opacity": 1
}]
```

```
"shape": {
  "type": "rectangle",
  "width": 30,
  "height": 20,
  "fill": "#5b9bd5"
},
"effect": [{
  "time": 0.6,
  "opacity": 0
},{
  "time": 0.9,
  "opacity": 1
}]
```

静态模板

```
"shape": {
  "type": "rectangle",
  "width": 60,
  "height": 20,
  "fill": "#5b9bd5"
}
```

动态模板

```
"effect": [{
  "time": 0,
  "opacity": 0
},{
  "time": 0.3,
  "opacity": 1
}]
```

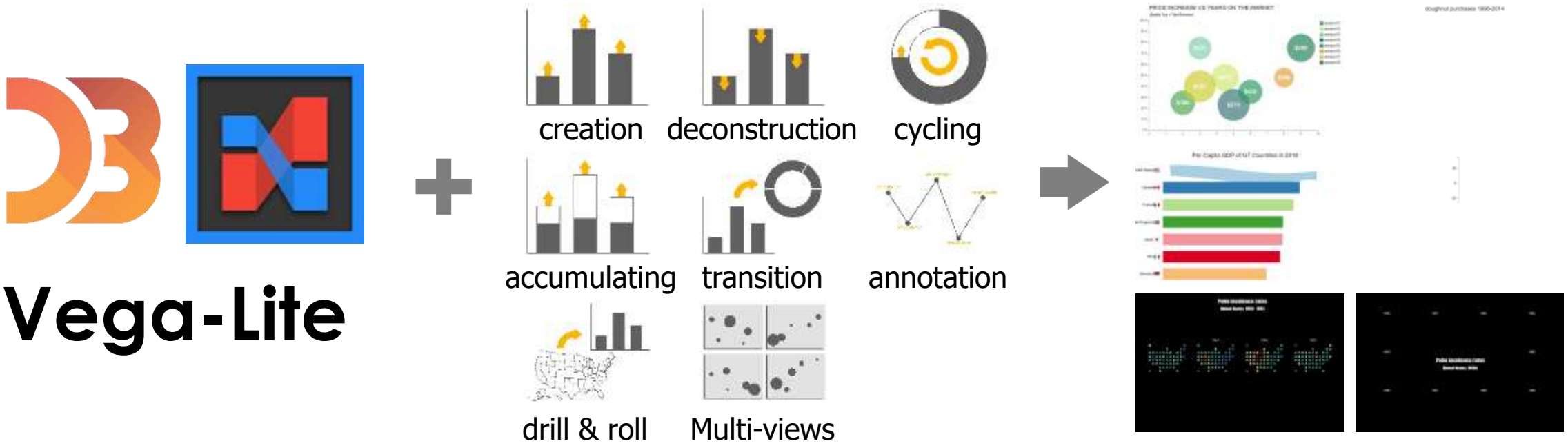
```
"shape": {"scaleX": 100},
"effect": [{"time": 0},{"time": 0.3}]
```

```
"shape": {"scaleX": 150},
"effect": [{"time": 0.3},{"time": 0.6}]
```

```
"shape": {"scaleX": 50},
"effect": [{"time": 0.6},{"time": 0.9}]
```

Canis – 评估（表现力）

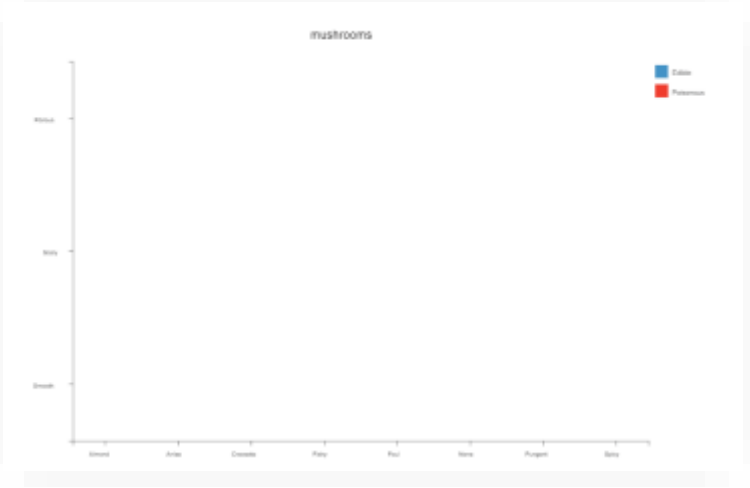
图表动画表现力由**可视化**和**动画过程**共同决定。因此在评估实验中分别采用：



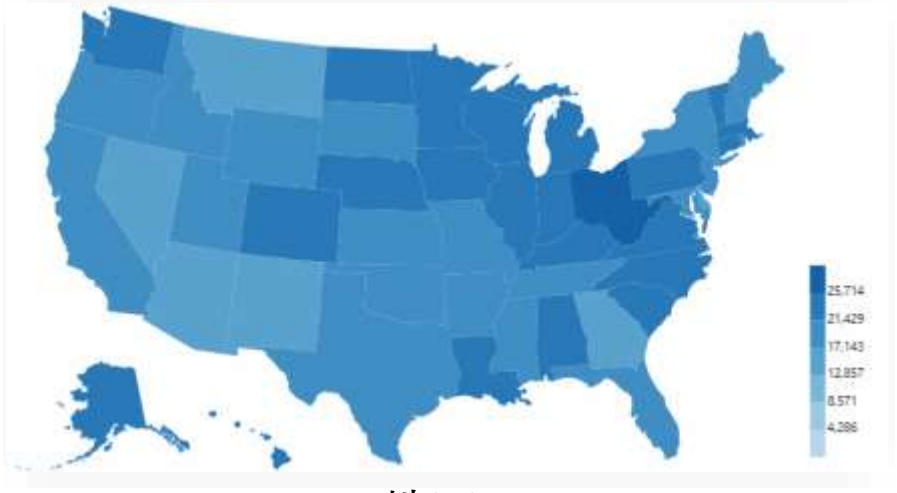
Amini et. Al, 2016

Canis – 评估（表现力）

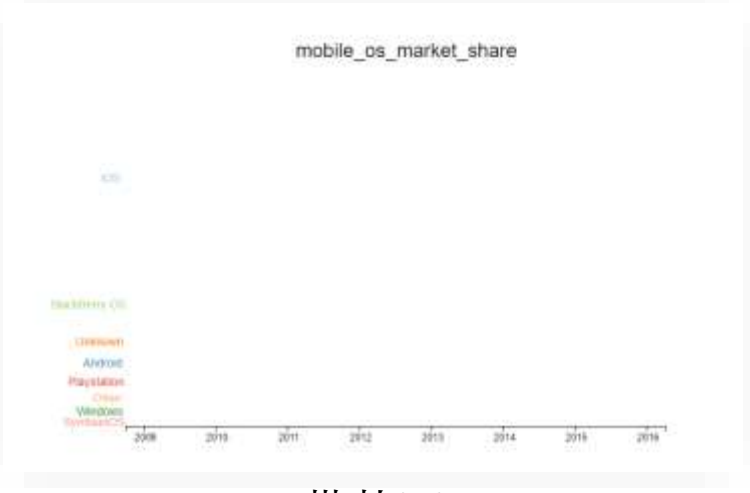
折线图



矩阵散点图

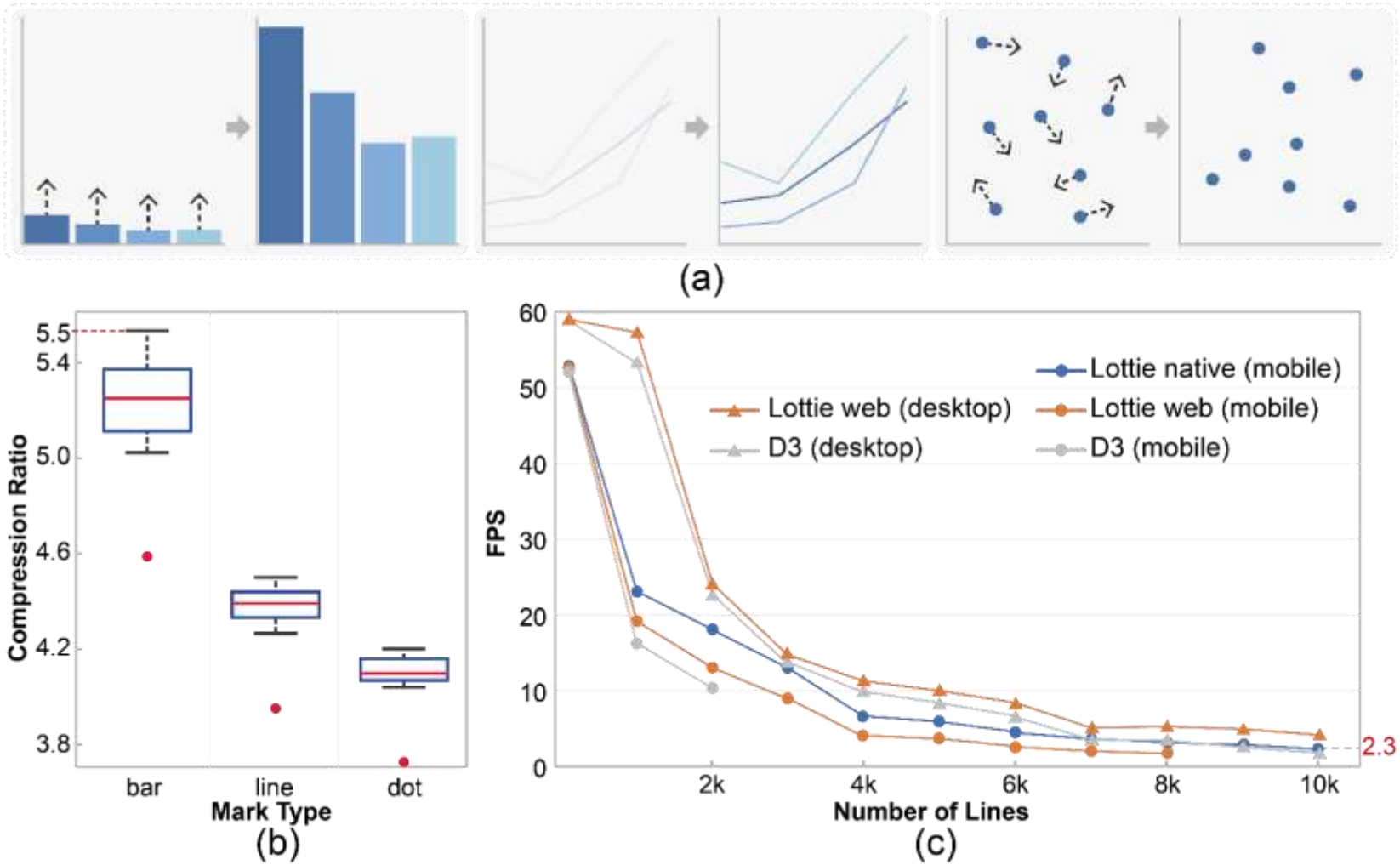


地图



带状图

Canis – 评估（渲染效率）

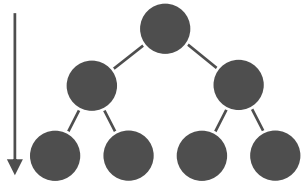


- 转译过程中使用可复用模板可极大的压缩输出动画体积
- 输出Lottie动画的渲染效率高于D3

Canis – 局限性

应用

动画可理解性差



自顶向下的构建顺序

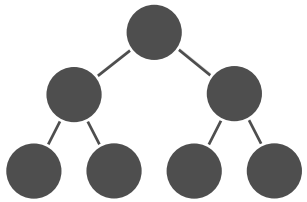
需要编码

- 直观的动画表现形式
- 数据驱动的动画构建过程优化

编译

```
"grouping": {
  "groupBy": "attr1",
  "grouping": {
    "groupBy": "attr2"
  }
}
```

局部更新



全局更新

语法

编码相同数据　定义时序相关性

- 语法扩展
- 编译机制优化

研究内容

CAST (CAnis STudio)：可视化动画交互式创作平台

Canis：数据驱动的可视化动画编程语言

语法扩展

编译优化

CAST – 目标

CAST

易理解的动画
表现形式

高效、直接的
构建方式

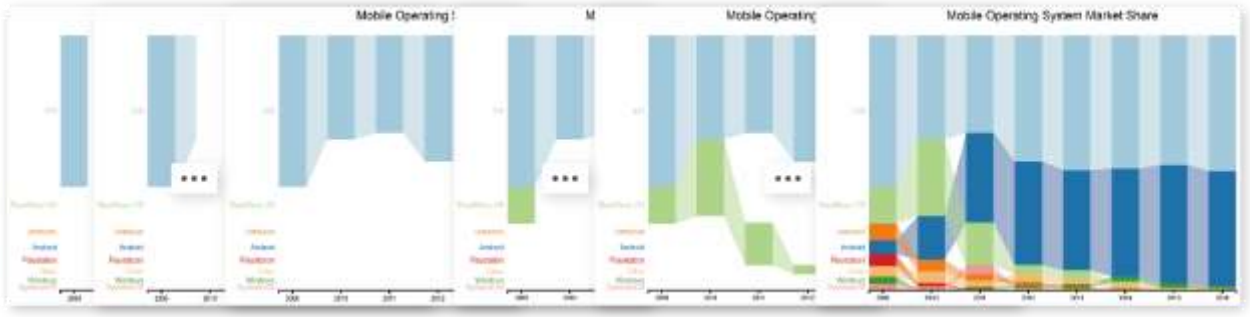
Canis

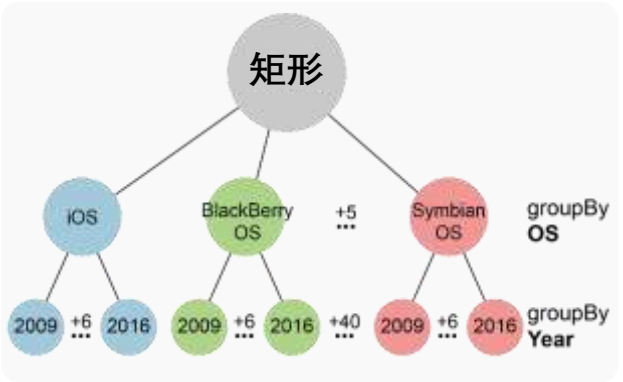
更高效的
增量式编译

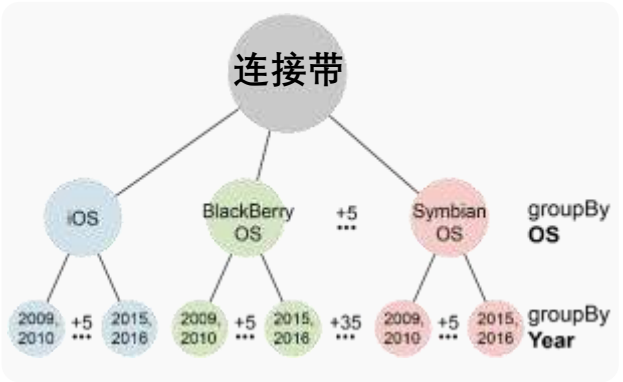
数据驱动
语法扩展

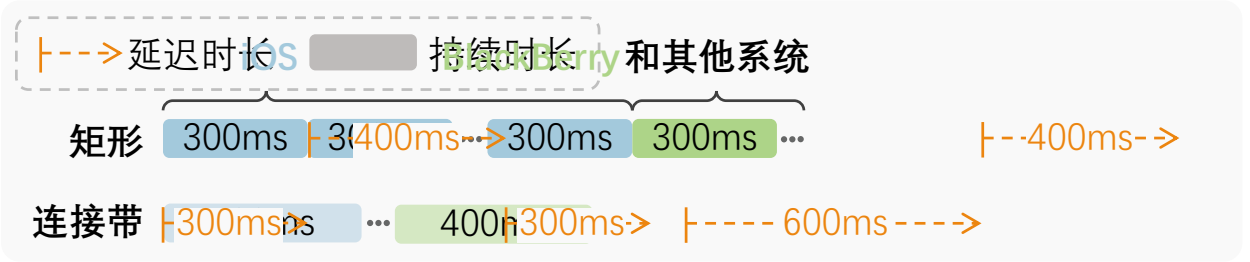
丰富图表动画

CAST – Canis语法扩展



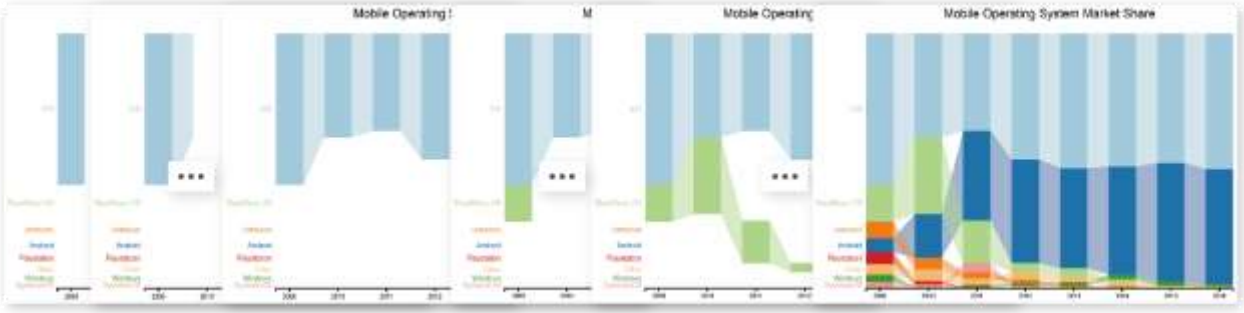


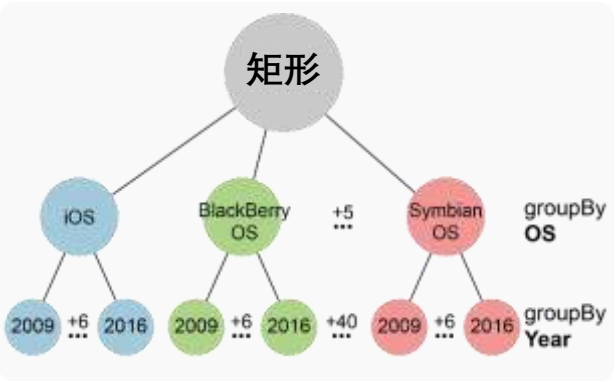


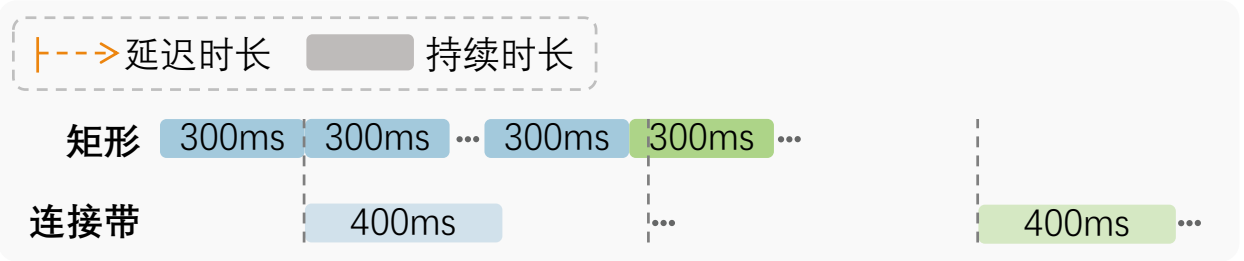



```
{
  "selector": ".rectangle",
  "grouping": {
    "groupBy": "OS",
    "timing": {
      "reference": "start after previous"
    },
    "grouping": {
      "groupBy": "Year",
      "timing": {
        "reference": "start after previous",
        "delay": 400
  }}},
  "effects": [{
    "type": "fade",
    "duration": 300
  }]
},{
  "selector": ".band",
  "timing": { "delay": 300 },
  "grouping": {
    "groupBy": "OS",
    "timing": {
      "reference": "start after previous",
      "delay": 600
    },
    "grouping": {
      "groupBy": "Year",
      "timing": {
        "reference": "start after previous",
        "delay": 300
  }}},
  "effects": [{
    "type": "wipe left",
    "duration": 400
  }]
}
```

CAST – Canis语法扩展



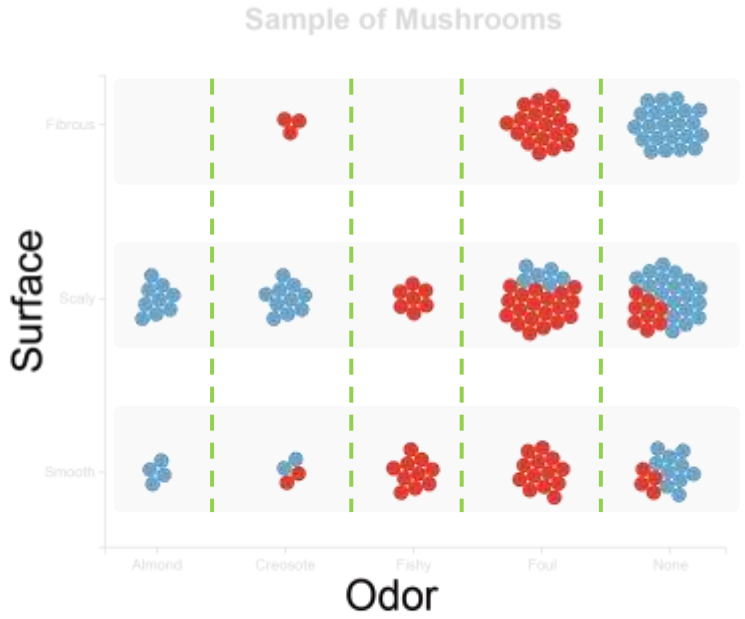




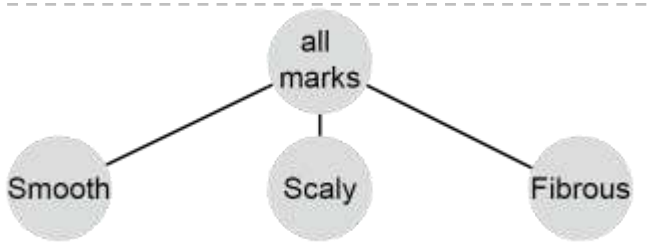

```
{
  "selector": ".rectangle",
"grouping": {
  "groupBy": "OS",
  "timing": {
    "reference": "start after previous"
  },
  "grouping": {
    "groupBy": "Year",
    "timing": {
      "reference": "start after previous",
      "delay": 400
}}},
  "effects": [{
    "type": "fade",
    "duration": 300
  }]
},{
  "selector": ".band",
  "timing": { "delay": 300 },
"grouping": {
  "groupBy": "OS",
  "timing": {
    "reference": "start after previous",
    "delay": 600
  },
  "grouping": {
    "groupBy": "Year",
    "timing": {
      "reference": "start after previous",
      "delay": 300
}}},
  "effects": [{
    "type": "wipe left",
    "duration": 400
  }]
}
```

```
{
  "id": "rect",
  "selector": ".rectangle",
"grouping": {
  "groupBy": "OS",
  "timing": {
    "reference":
      "start after previous"
  },
  "grouping": {
    "groupBy": "Year",
    "timing": {
      "reference":
        "start after previous"
}}},
  "effects": [
    "type": "fade",
    "duration": 300
  }]
},{
  "selector": ".band",
  "timing": {
    "reference":
      "start after previous"
  },
  "align": {
    "target": "rect",
    "type": "element"
  },
  "effects": [{
    "type": "wipe left",
    "duration": 400
  }]
}
```

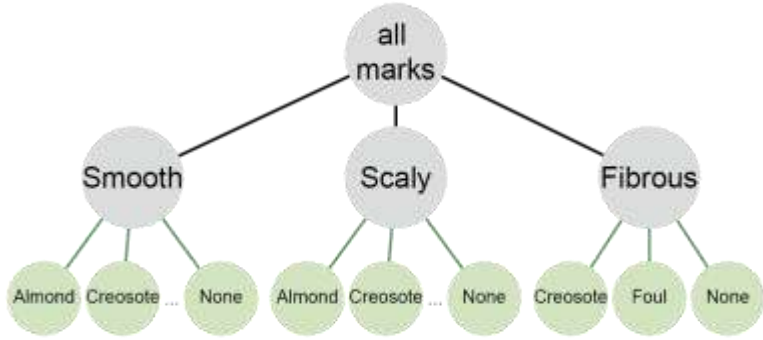
CAST – Canis编译优化




```
"grouping": {
  "groupBy": "Surface",
  "timing": {
    "reference": "start after previous"
}}
```




```
"grouping": {
  "groupBy": "Surface",
  "timing": {
    "reference": "start after previous",
    "delay": 400
  },
  "grouping": {
    "groupBy": "Odor",
    "timing": {
      "reference": "start after previous"
}}}
```



CAST – 设计原则

DP1: 精确直观的视觉表现形式

引入视觉规范，通过使用视觉组件及其布局方式表述动画参数，实现对动画过程的精确直观展示

DP2: 简单高效的构建过程

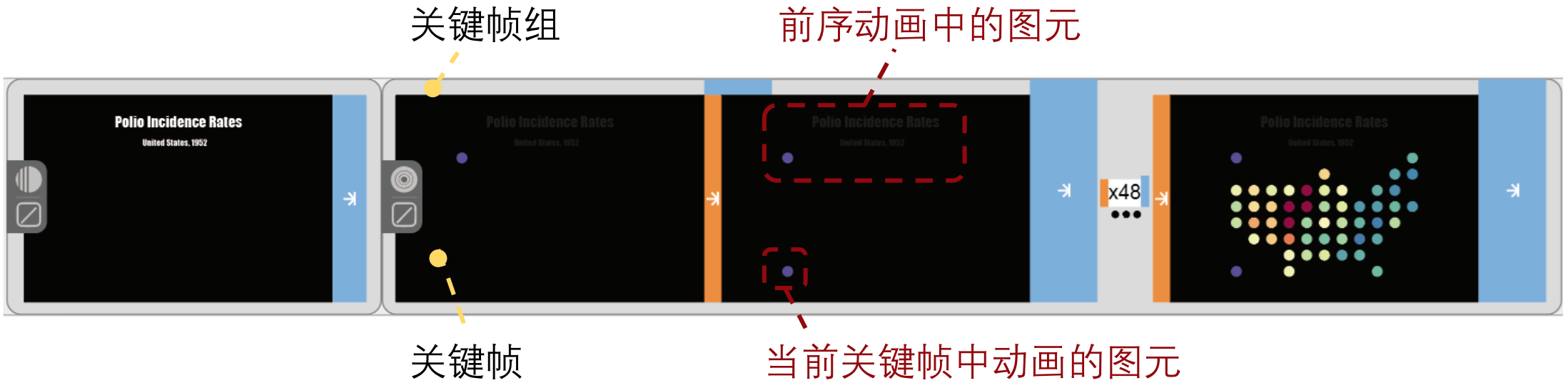
通过借助数据驱动的推荐及补全算法，简单高效的完成关键帧及动画序列构建

DP3: 直接的交互方式

平衡创作过程中直接交互和参数面板的功能

CAST – 视觉规范

DP1: 精确直观的视觉表现形式



关键帧与**关键帧组**以故事板（Storyboard）的形式展示了动画过程中的关键步骤。

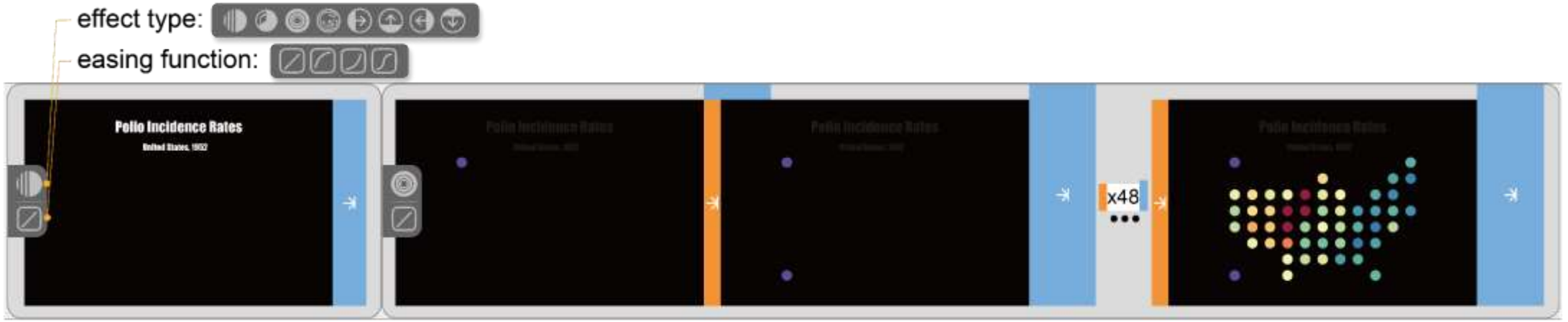
CAST – 视觉规范

DP1: 精确直观的视觉表现形式

图元单元动画之间的时间关系由**时间条**（持续时间、延迟时间）和关键帧（组）的**布局**共同决定

CAST – 视觉规范

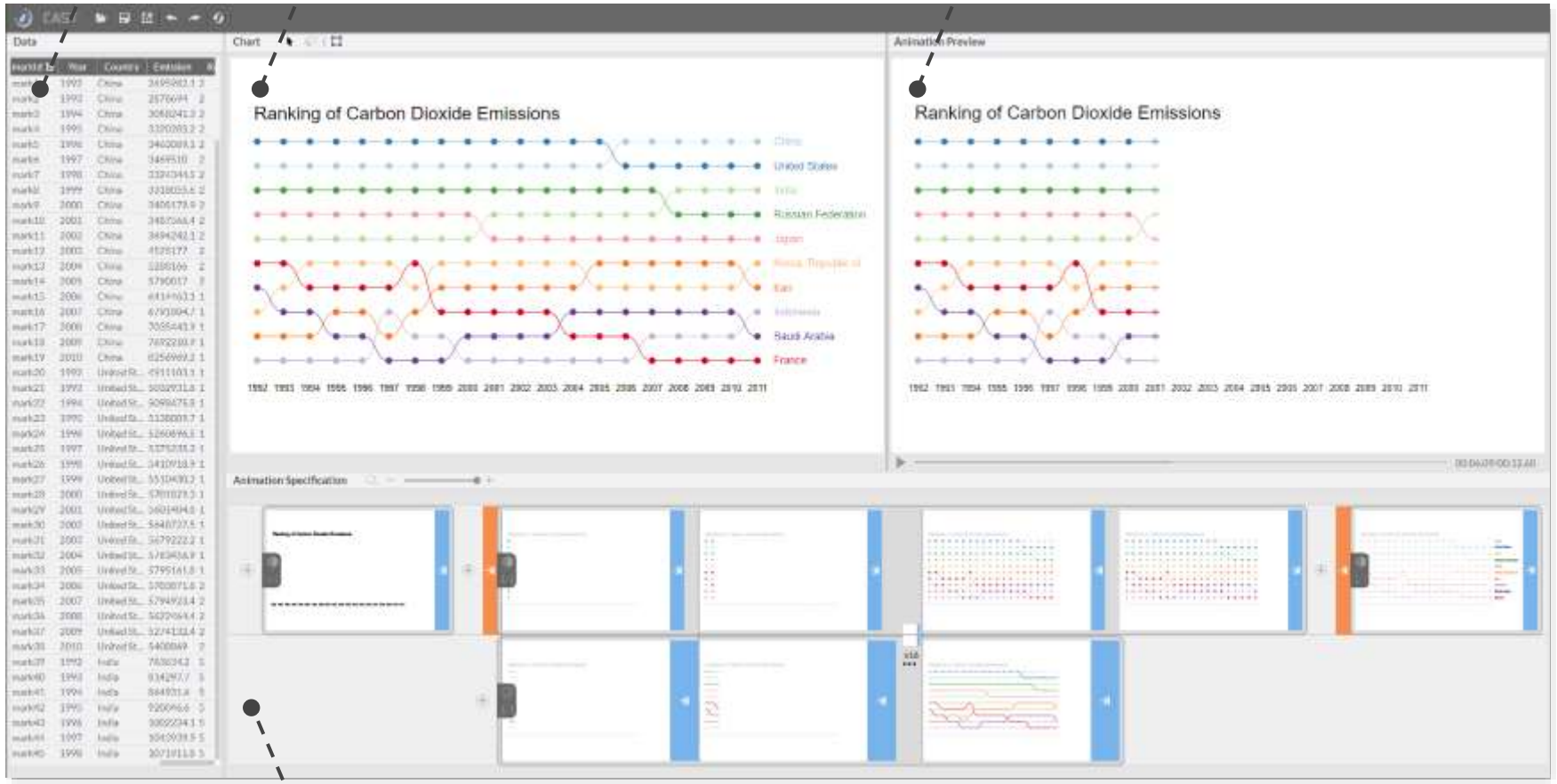
DP1: 精确直观的视觉表现形式



CAST支持8类预定义**动画效果**和4类**缓动类型**

CAST – 系统总览

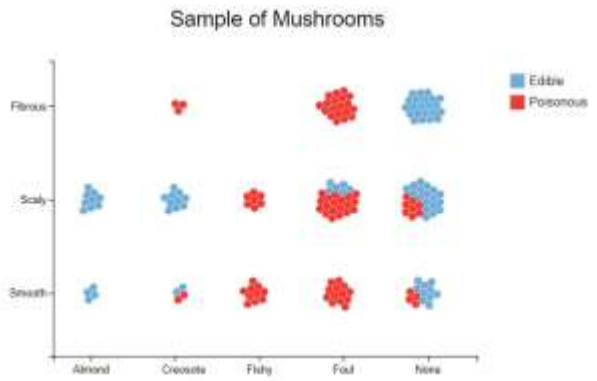
数据面板　　输入图表面板　　动画预览面板



动画创建面板

CAST – 基本交互机制

DP3: 直接的交互方式

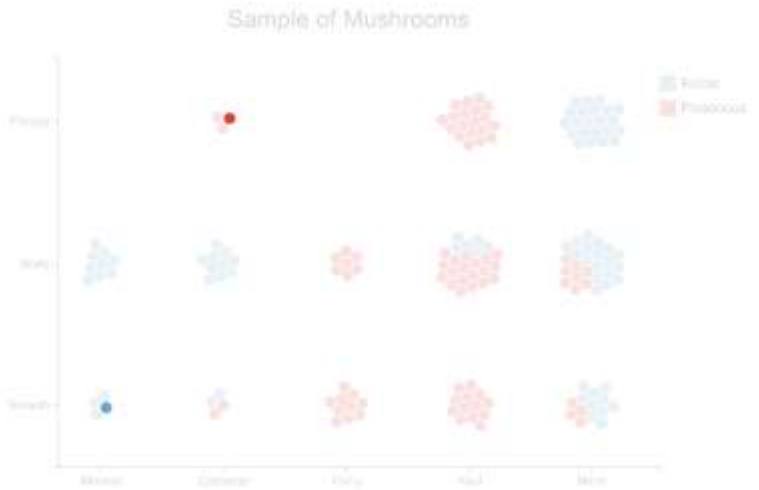


Input Chart

图元选择[illegible]自动补全

CAST – 智能化交互设计（关键帧自动补全）

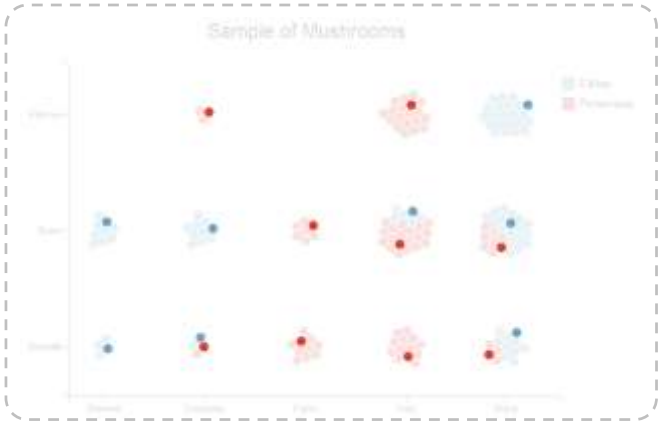
DP2: 简单高效的构建过程



视觉编码:

- X位置: Odor
- Y位置: Surface
- 颜色: IsEdible

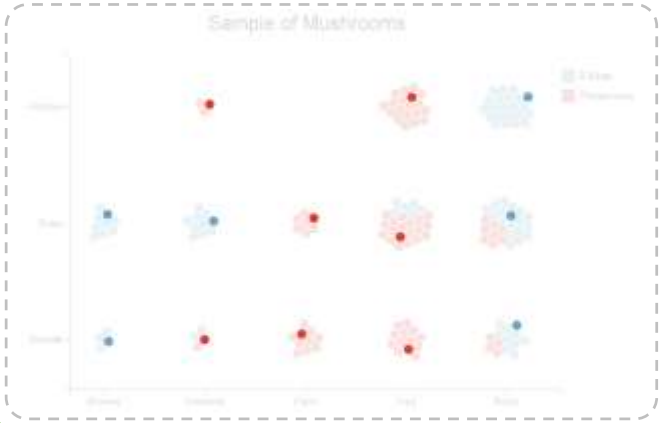
方案1:



Partition by:

- Odor
- Surface
- IsEdible

方案2:



Partition by:

- Odor
- Surface

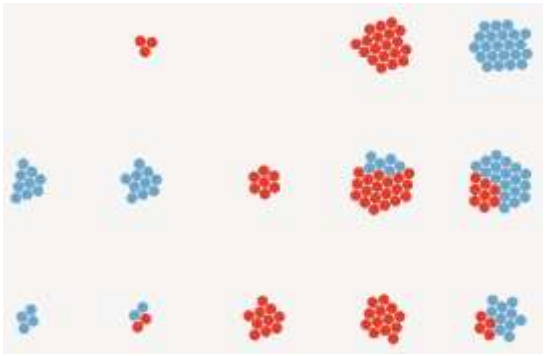
视觉通道的有效性排序:

位置 > 颜色

CAST – 智能化交互设计（关键帧推荐）

DP2: 简单高效的构建过程

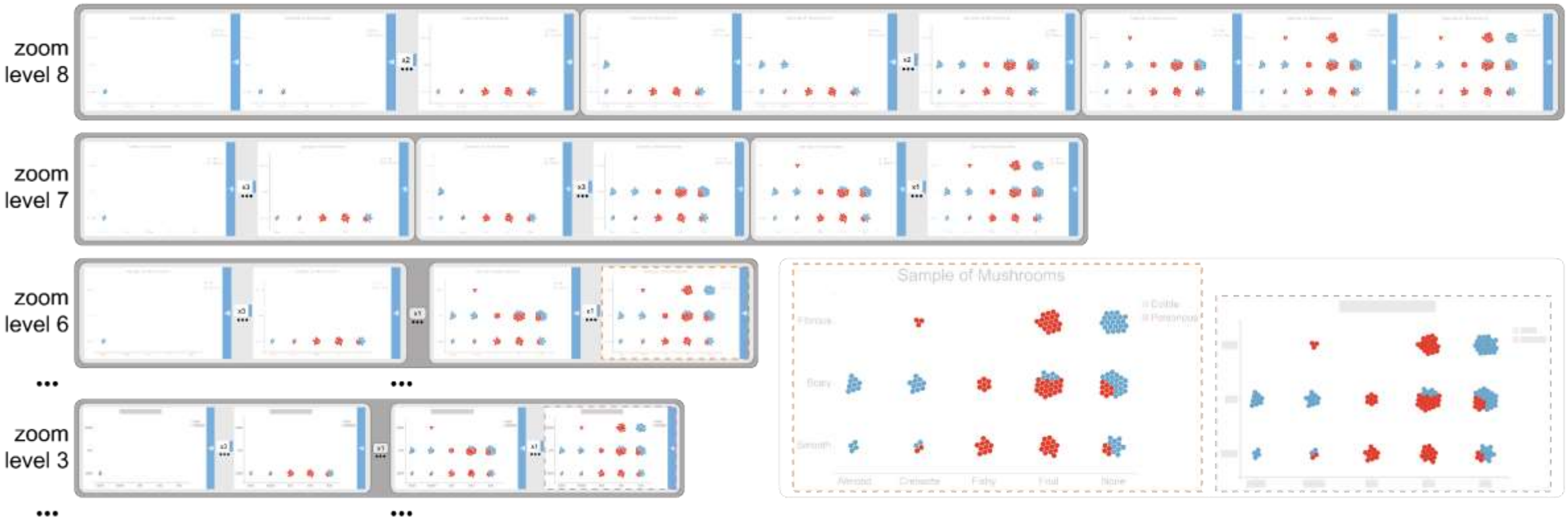
已选图元单元



全部图元

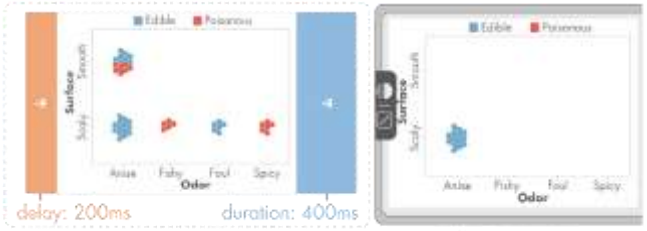
CAST – 智能化交互设计（语义缩放）

DP2: 简单高效的构建过程

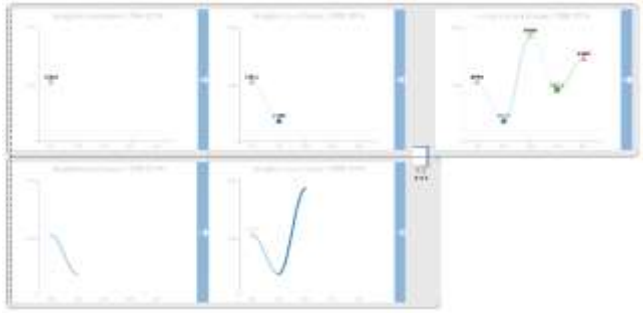


CAST – 评估（理解视觉规范）

验证CAST中视觉规范的**易学习性**和**可理解性**。



视觉组件含义



关键帧（组）布局

教学

选择与视觉规范的描述一致的动画

训练（3）/正式实验（4）

CAST – 评估（复现图表动画）

验证CAST系统的**易学习性**和**易用性**。

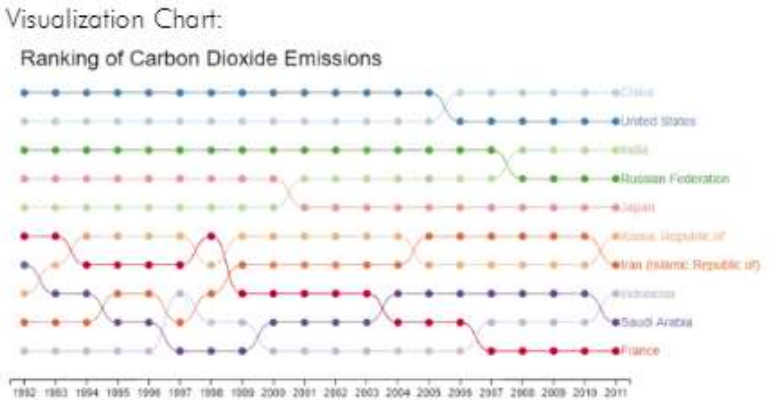
图元选择

关键帧序列构建

关键帧时序调整

关键帧组时序调整

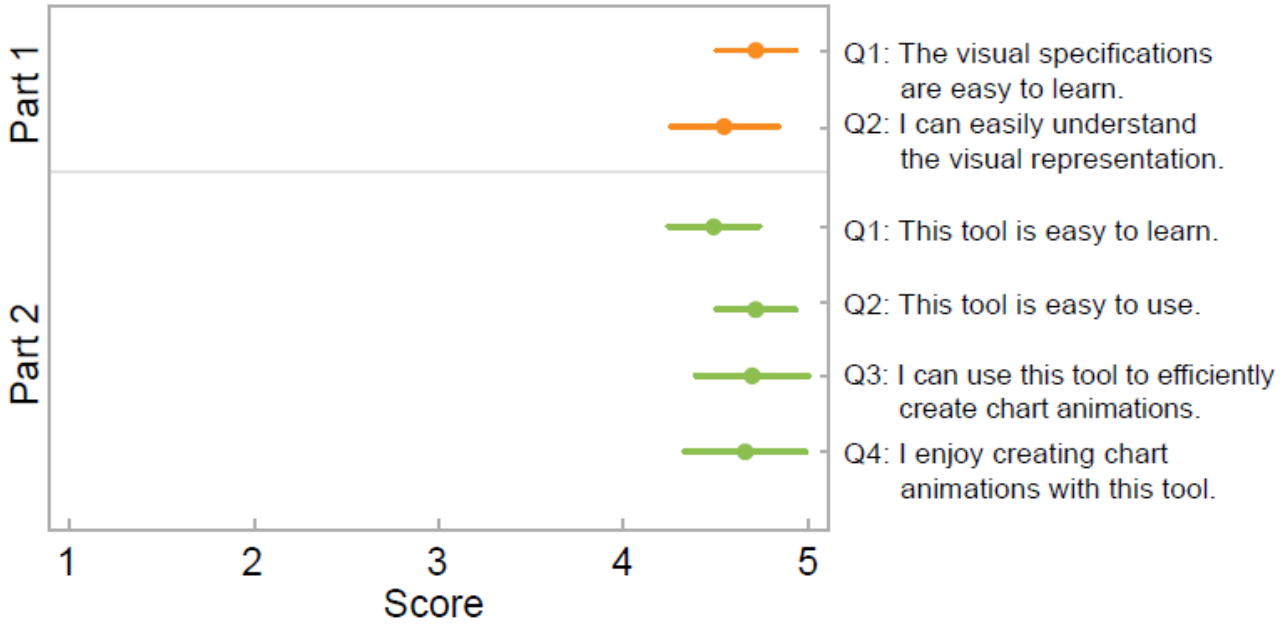
教学



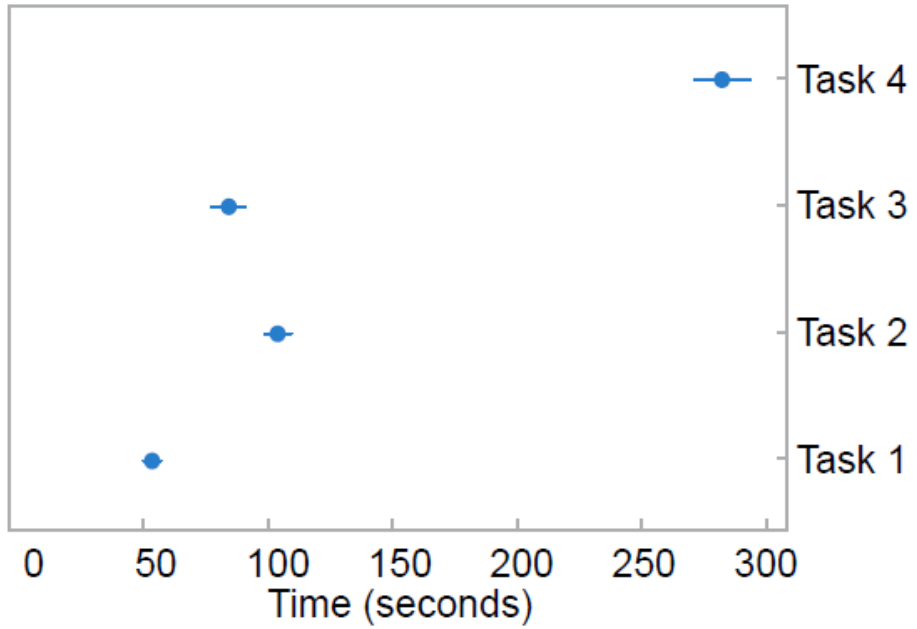
动画复现

训练（3）/正式实验（4）

CAST – 评估（实验结果）



用户反馈



复现动画完成时间

目录

TOC

总结

提出了一种数据驱动的可视化动画高级编程语言 —— Canis

Canis让设计者可以用简洁易懂的声明式语法，以数据驱动的方式为各类可视化图表创建丰富的动画。在保证了有效性和表现力的前提下，提高了创作效率。

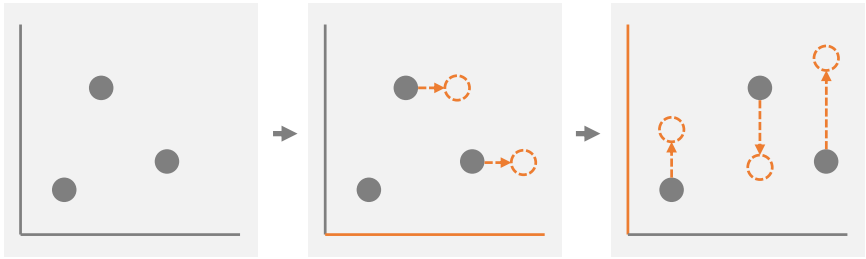
提出了一种基于视觉规范和自动补全算法的交互式可视化动画创作平台 —— CAST

通过以视觉组件的形式描述和操控动画过程，并采用数据驱动的自动补全算法，CAST让新手用户可以简单快捷构建丰富的可视化动画。

工作展望

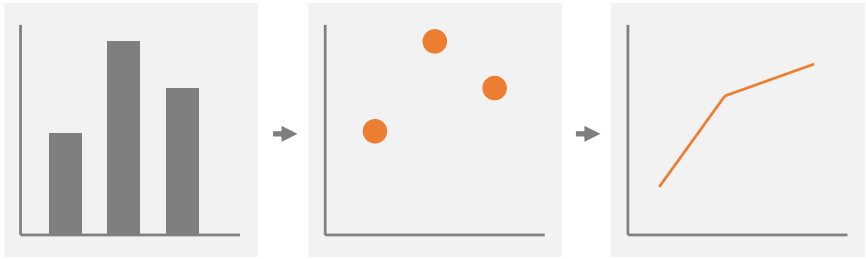
01 **支持多阶段动画的描述与生成**

扩展语法、视觉规范、交互方式以支持对复杂动画过程的分阶段描述和创作，进一步增强系统表现力。



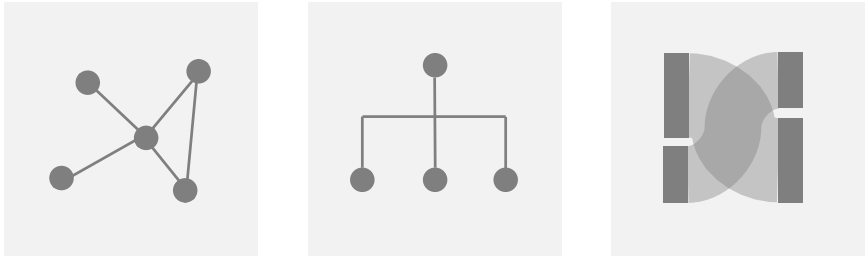
02 **支持多张图表之间的过渡动画**

扩展语法，增加多图表过渡动画声明规则，并依此改进交互方式以支持对该类动画进行直观创作。



03 **支持更多的可视化图表及动画类型**

对表示数据之间联系的图元的动画参数进行自动计算，以丰富对树、关系图、桑基图等的动画创作。



感谢大家聆听！

Canis:

https://chartanimation.github.io/canis/ (主页)
https://github.com/ChartAnimation/canis-compiler (github)

CAST:

https://chartanimation.github.io/cast/ (主页)
https://github.com/ChartAnimation/cast (github)