Optimal Fitting of Planar Curves to Prescribed Constraints

Julian Asamer, Julian Brunner

January 28, 2014

Introduction

- vector graphics are widely used
- 2D vector graphics primitives are curves
- design process of curves is very important
- designing curves with current tools can be frustrating
- research has been done in different directions with little impact
- our objective: improved curve design process

The Curve Design Process

- obtain source curve
- extract properties from the source curve
- provide them to the software
- the software constructs a curve

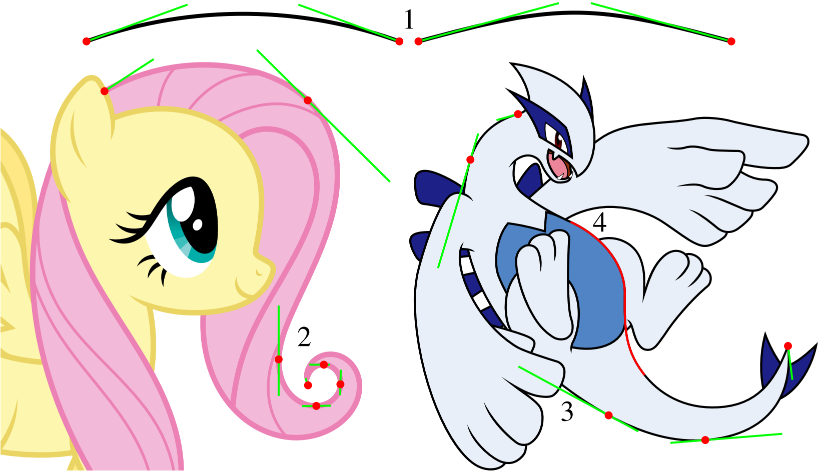
Usability Criteria for Curve Design Tools

- description language := specification language + fairness measure
- specification language should be
    - expressive
    - easy
    - efficient
- fairness measure should
    - ensure smoothness
    - select minimal curves
- ability to derive curves from descriptions

Bézier Splines

- most popular curve type
- specify point and velocity at start and end of each segment
- fairness measure ‘chooses’ unique cubic polynomial
- most usability criteria fulfilled sufficiently
- usability problems
  - curvature continuity is not guaranteed
  - lack of expressiveness of the specification language

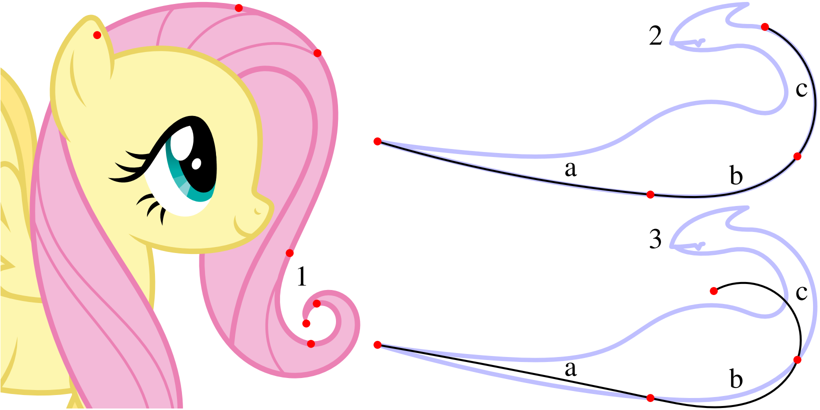
Bézier Splines - Examples



Spiro Splines

- consist of parts of the Euler spiral
- construct interpolating splines
- fairness measure guarantees curvature continuity
- usability problem: lack of expressiveness of the specification language

Spiro Splines - Examples



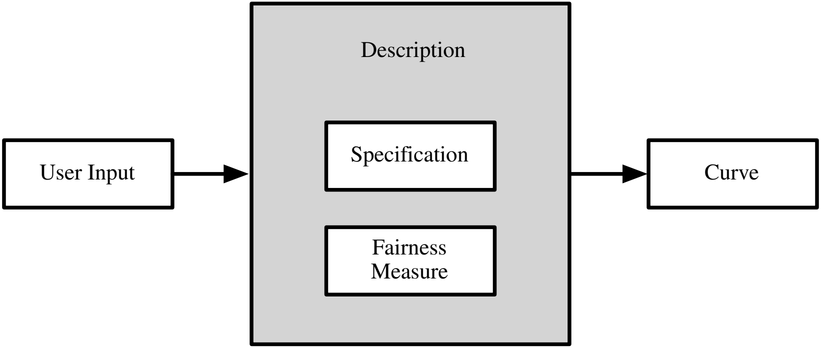
Definitions

$\phi : [0,1] \to \mathbb{R}^2$	point
$\sigma : [0,1] \to \mathbb{R}$	speed
$\lambda : [0,1] \to \mathbb{R}$	covered arc length
$\delta : [0,1] \to \mathbb{R}$	direction
$\chi : [0,1] \to \mathbb{R}$	curvature

Description-Based Curves - Motivation

- description language has huge effect on usability
- should not be based on low-level mathematical aspects
- do not impose limitations on specification language prematurely
- we need a good specification language and fairness measure

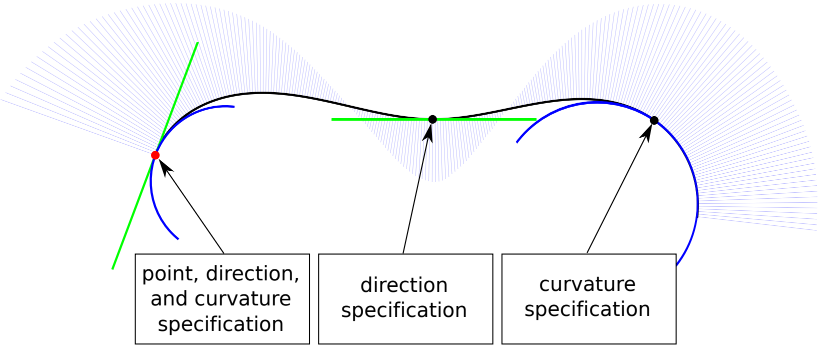
Description-Based Curves



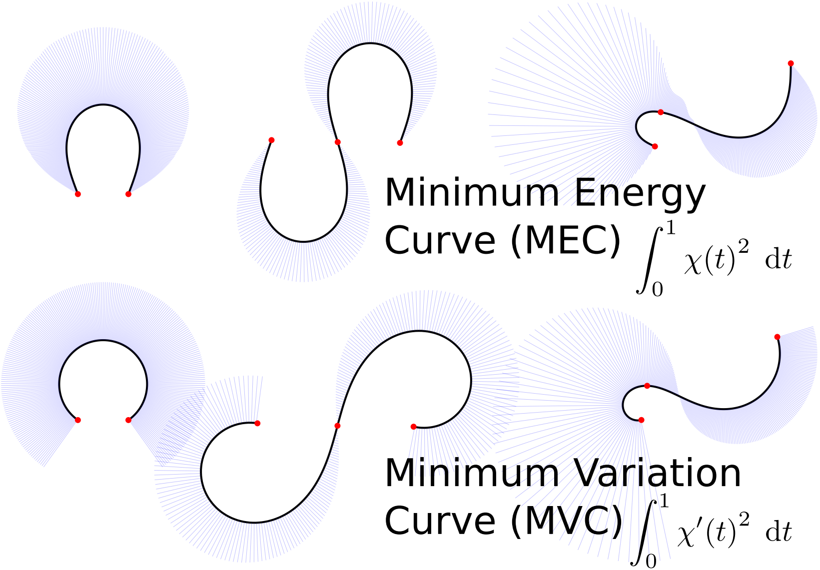
Specification Language

- decoupled from underlying curve model
- allows specification of geometric properties
- point, direction, curvature
- any combination, any position
- positioning via fraction of arc length of curve
- total curve length specified as well

Specification Language - Example



Fairness Measure



Minimum Energy Curve (MEC) $\int_0^1 \chi(t)^2 \; \mathrm{d}t$

Minimum Variation Curve (MVC) $\int_0^1 \chi'(t)^2 \; \mathrm{d}t$

Curve Derivation

- turn descriptions into actual curves
- approach: nonlinear optimization on polynomial splines
  - nonlinear optimization: flexible, allows for experimentation
  - polynomial curves: simple, versatile
  - polynomial splines: avoids polynomials of high degree

Optimization Problem

- using
  - objective function $f : \mathbb{R}^n \to \mathbb{R}$
  - constraint function $g : \mathbb{R}^n \to \mathbb{R}^m$
  - constraint bounds $g_l, g_u \in \mathbb{R}^m$
- try to find an $x^* \in \mathbb{R}^n$ such that

$$x^* = \operatorname*{argmin}_{x \in \mathbb{R}^n} f(x)$$

$$g_l \leq g(x^*) \leq g_u$$

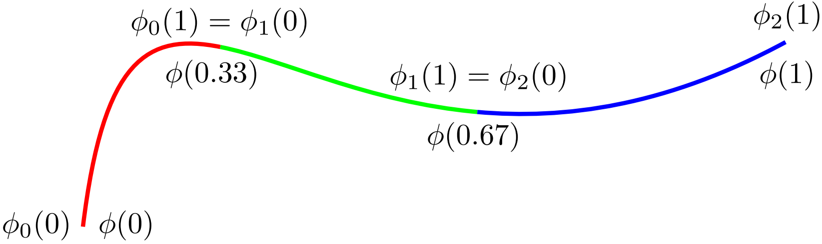
- smooth optimization (interior point method)

Finding Curves as Optimization Problem

- segment polynomials

$$\phi_i(t) = \sum_j a_{i,j} \cdot t^j$$

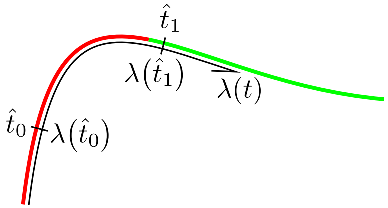
- segmentation



- optimization domain: coefficients $a_{i,j}$

Covered Arc Length Function

- $\lambda(t)$ relates parameter values to covered arc length



- example: find $\hat{t}$, such that $\lambda(\hat{t}) = 0.4$ and $\delta(\hat{t}) = 15^\circ$
- requires finding $\lambda^{-1}(\hat{t})$
- result determines segment for specification item
- error terms unsuitable for smooth optimization
- further complications: error terms for arc length and fairness measure

Constant Speed

- solution: add constant speed as optimization goal
- error term

$$\int_0^1 \left(\sigma(t) - \hat{\lambda}\right)^2 \, \mathrm{d}t$$

- covered arc length function approximately linear: $\lambda(t) \approx \hat{\lambda} t$
- curve's total arc length approximately $\hat{\lambda}$
- error terms for specification items become trivial

Continuity Connections

- require G^2 continuity
- each segment is smooth, resulting spline may not be
- ensure G^2 continuity at segment connection points
- error terms

$$\phi_i(1) - \phi_{i+1}(0)$$
$$\phi_i'(1) - \phi_{i+1}'(0)$$
$$\phi_i''(1) - \phi_{i+1}''(0)$$

Fairness Error

- MVC fairness measure
- error term

$$\int_0^1 \chi'(t)^2 \, \mathrm{d}t$$

Overall Curve Derivation Process

- construct optimization problem from error terms
- compute coefficients with optimization solver
- build result curve from coefficients

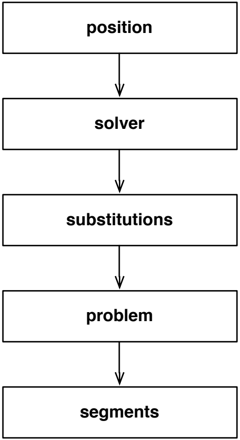
Architecture

- programming language: C#
  - platform independent
  - interfaces with native code
  - UI frameworks available
- libraries
  - Ipopt: numeric nonlinear optimization
  - CasADi: automatic differentiation of terms
  - GTK#: UI library

Subsystems

- Kurve: UI prototype
- Kurve.Curves: optimization of curves
- Wrappers.Casadi and Wrappers.Casadi.Native: APIs for CasADi and Ipopt

Optimization Steps



Demo

Conclusion

- description-based curves look promising
- most problems of prototype caused by optimization
- future work
  - research more constructive approach for building curves
  - add even more expressiveness to description language
  - more research on soft specifications
  - usability study with actual designers
  - Inkscape integration