4
Minimum Variation Curves

In this chapter we introduce the Minimum Variation Curve (MVC) as a strong candidate for use in CAD applications and as a building block in the construction of MVN and MVS. The MVC has superior fairness, handles constraints gracefully, and forms circular arcs as a general case of free-form curve design.

The MVC is computed using numerical optimization and finite element techniques to find the curve that minimizes a fairness functional measuring the variation of curvature. In this chapter we discuss the techniques used in this calculation and present trade-offs in the decisions made.

4.1 Curve Specification Through Constraints

Curves are specified with a sequence of geometric constraints. The points to be interpolated in sequence are specified, optionally augmented with constraints on tangents, and/or curvatures. Figure 4.1 illustrates the 3 basic constraint types: positional, tangential, and curvature, and shows that curvature discontinuities may also be specified. In Fig. 4.1① we see how three point constraints form a circular arc. Figure 4.1② illustrates how a tangent constraint is combined with two point constraints to specify a curve. Figure 4.1③ adds a curvature constraint, providing further control over the curve shape. Finally, Figure 4.1④ inserts a curvature discontinuity to allow circular curve segments to be formed.

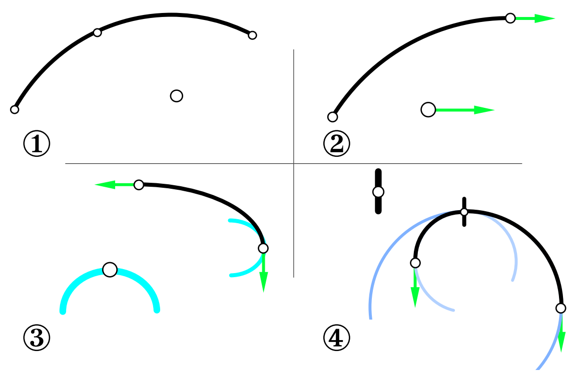


Figure 4.1. Specification Through Geometric Constraints.
① three point constraints. ② a point and a point-tangent constraint. ③ a point-tangent and a point-curvature constraint. ④ a curvature discontinuity.

4.2 Functionals for Minimization

In designing the MVC, our choice of functional for minimization was made based on the need for high quality curves and intuitive interactive behavior. The quality or *fairness* of curves has been studied extensively and shown to be closely related to how little and how smoothly a curve bends. The cubic spline is a linear approximation to an idealized thin beam at minimum strain energy passing through a series of frictionless, swiveling supports. The nonlinear curve modeling a thin beam is known as the minimum energy curve or MEC and is characterized by *bending the least* while passing through a given set of points. The MEC functional minimizes the arc length integral of curvature squared

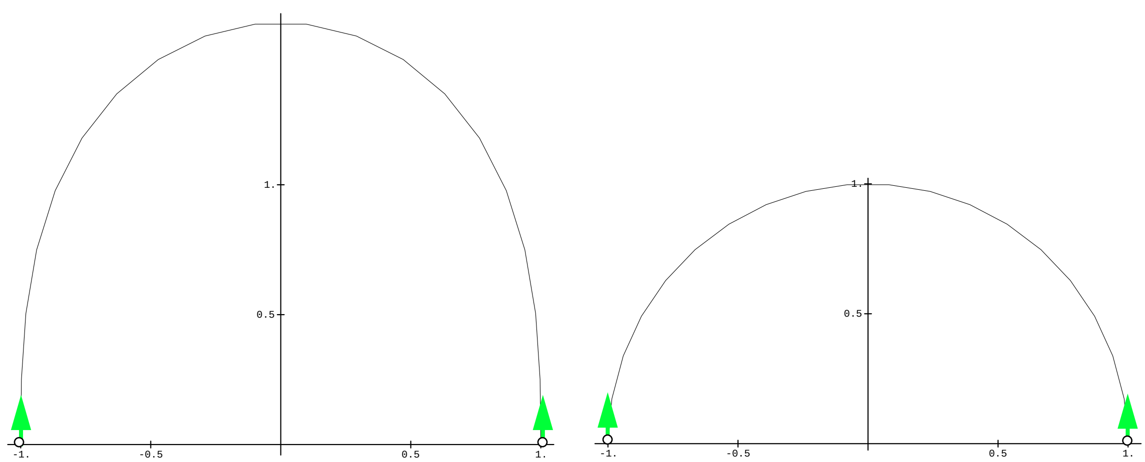
$$\int \vec{\kappa}^2 ds. \tag{4.1}$$

Figure 4.2① provides an example of the shape formed by an MEC for a simple set of constraints.The MEC has a few drawbacks. It does not readily form circular or helical shapes without imposition of arc length constraints, and it is not stable in all configurations. If the angular difference between tangents at adjacent vertices becomes greater than π, then the curve bulges out, growing to infinity.

In contrast to the MEC which bends as *little* as possible, the minimum variation curve (MVC) bends as uniformly or as *smoothly* as possible while passing through a series of points. The uniformity of bending is measured by the magnitude of the rate of change of curvature, minimizing the functional

$$\int \left(\frac{d\vec{\kappa}}{ds}\right)^2 ds. \tag{4.2}$$

Figure 4.2① illustrates the shape of the MVC. Note that the MVC forms a circular arc in response to these symmetric constraints. The MVC is more stable, changing shape continuously with continuous changes in constraints. The MVC also has the property that it forms circular arcs when constraints allow.



MEC ① **MVC** ②

Figure 4.2. The Wicket.

The MEC and MVC resulting from two point-tangent constraints. The MEC has zero curvatures at its end points with a higher interior curvature. The MVC forms a semicircle; curvature is constant throughout.

4.2.1 Functionals for Space Curves

In this section we discuss potential modifications of functionals (4.1) and (4.2) when applied to space curves; though these modifications have not been implemented, they are included to complete the discussion of functionals. When formulating a functional for modeling space curves (non-planar curves), torsion becomes a factor. In the context of the traditional MEC functional based on strain energy, a torsion term must be added to the functional to account for twisting,

$$\int \kappa^2 + \tau^2 ds. \tag{4.3}$$

We are not aware of any other work on curves designed with this functional(4.3). Analogously, the *space MVC* functional becomes

$$\int \left(\frac{d\kappa}{ds}\right)^2 + \left(\frac{d\tau}{ds}\right)^2 ds. \tag{4.4}$$

Note that in contrast to (4.2) this expression is composed of the *scalar*-valued derivatives of curvature and torsion. The result is a functional that evaluates to zero for helices, which of course include circles. In comparing (4.2) to (4.4) note that in the former we use the *vector*-valued derivative of curvature. (4.2) indirectly captures the torsion term since for a space curve the derivative of the curvature vector is nonzero even if its magnitude is constant. The MVC and space MVC functionals are not equivalent, and the latter is superior in its ability to produce fair space curves. As we see in section 4.11, the MVC functional produces curves with zero torsion at their end points, while the space MVC functional will produce curves with constant torsion at their end points. Because of the substantially increased complexity in computing the arc length derivative of torsion, we have chosen to use the MVC functional with its lower complexity and indirect measure of torsion. A thorough investigation of space MVC is slated for future research.

4.3 Scale-Invariant MVC

MVC are invariant under rigid body transformations and uniform scaling. The *value* of the MVC functional, however, changes with a change of scale. The precise rate of change may be determined most easily by incorporating a scale factor into the parametric version of the functional (section 4.7) and factoring it out. We find that as the scale increases by a factor of γ, the functional value is reduced by a factor of $1/\gamma^3$. The MEC functional has a similar sensitivity to scaling, in its case the reduction is by a factor of $1/\gamma$. We may modify the MVC functional to create an MVC-like functional whose value is independent of scale (SI-MVC). In designing this new functional we must be sure to preserve the desirable properties of the MVC functional; its characteristic shapes, parameterization independence, and discretization independence. By adding terms which are composed solely of geometric measures we guarantee parameterization and discretization independence. To insure that the SI-MVC functional has the same characteristic shapes we must guarantee that if the standard MVC functional evaluates to zero then the SI-MVC functional must also evaluate to zero. Taking these factors into consideration, we have designed a functional that incorporates an arc length term to offset the scaling factor,

$$\left(\int ds\right)^3 \int \left(\frac{d\kappa}{ds}\right)^2 ds. \qquad \textit{the SI-MVC functional} \qquad \mathbf{(4.5)}$$

We have found that this function is scale independent, but has much slower convergence than the standard MVC functional. This functional does, however, allow us to investigate the characteristic shape of various curve topologies which cannot be investigated with the MVC functional without interpolation constraints. While curves that are topologically equivalent to a circle actually converge to a circle because the functional tends towards zero, other curves such as a figure-eight, expand to infinity, taking advantage of the $1/\gamma^3$ factor. However, when we use the SI-MVC functional (4.5), the figure-8 curve simply forms a lemniscate, see Figure 4.15 on page 91.

4.4 Local Control and Smoothing

In this section we explore what modifications of the MVC functional are needed to support variable locality and a facility for smoothing an existing shape. As described thus far, the shape of a curve is strictly a function of its geometric constraints and of the MVC functional. If constraints were removed or changed, the curve would change shape in response. If all constraints are removed, the curve will seek its *zero* or *reference shape.* The reference shape is the shape that the curve must form in order for the function to evaluate to zero. As it stands, the reference shape of an MVC is a circular arc. In order to allow for the specification of a non-circular reference shape, the MVC functional may be modified to

$$\int \left(\frac{d\vec{\kappa}}{ds} - \frac{d\vec{\kappa}_0}{ds}\right)^2 ds. \qquad \mathbf{(4.6)}$$

In (4.6) the term $\frac{d\vec{\kappa}_0}{ds}$ is the curvature derivative map of the curve's reference shape; we see that this new functional is zero when the curve forms the reference shape. In Figure 4.3① we provide an example of this type of deformation, in the figure, the reference shape is labeled "initial curve".

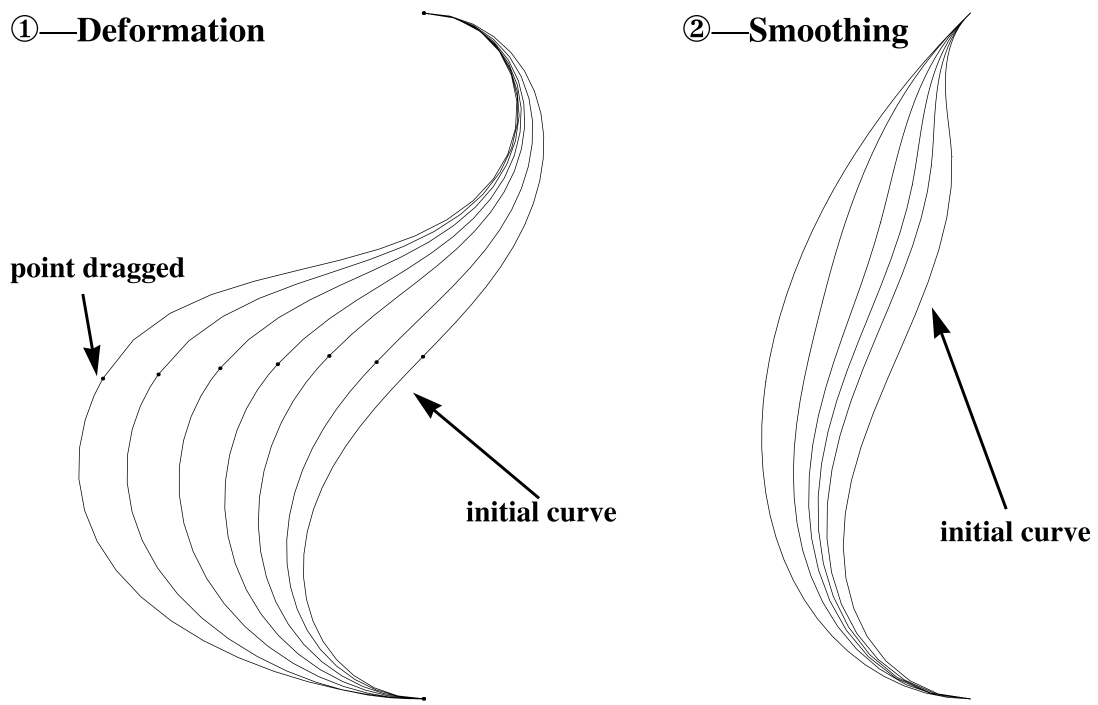


Figure 4.3. Curve Deformation and Smoothing.
① a curve is deformed from its initial reference shape by dragging a point near its middle.
② a curve is gradually smoothed from its initial shape to a shape defined solely by tangent constraints at its end points.

We now discuss how this modified functional may be used to achieve variable locality. Strictly local control can be achieve with the original functional. Modifications can be restricted to a region of the curve by fixing the position, tangent, and curvature at the end points of a region. To make modifications of larger scope, it is necessary to use the shape of the curve to be modified as its own reference shape. We may now remove the constraints that were used to define the curve's shape, and the curve will not change shape because (4.6) is minimized. We may now apply new constraints to the curve, deforming it from its reference shape.

Our new functional (4.6) may be further modified to allow for simple smoothing or relaxation operations. If we multiply the reference shape terms in (4.6) by a scale factor, it is possible to continuously vary the curve from its reference shape to its shape as defined by geometric constraints alone. The modified functional is

$$\int \left(\frac{d\vec{\kappa}}{ds} - \alpha\frac{d\vec{\kappa}_0}{ds}\right)^2 ds.$$

By varying α between 1.0 and 0.0 we vary the curve between its reference shape and its shape defined solely by geometric constraints. In Figure 4.3 we see examples of deformation from a reference shape and of the smoothing of a curve from it reference shape to its shape as defined by the constraints at its end points.

4.5 Representation

Starting with this section we begin describing our method for the computation of a minimum variation curve. We cast the problem as a nonlinear optimization / finite element problem. Curve representation plays an important role in making the MVC useful to applications. The curve is broken into a series of parametric polynomial elements that satisfy the given geometric constraints, and join with G^2 continuity. Among polynomials, quintic elements are most suitable because they have sufficient descriptive power to simultaneously satisfy the constraints on position, tangent direction, and curvature.

In considering alternatives, we examined lower order and rational polynomials. Using lower order polynomials, it is either difficult or impossible to constructively satisfy a specification, and little savings is gained because the majority of the complexity and cost is in the evaluation of the functional and its partial derivatives. For example, consider using a pair of cubic segments to replace a single quintic segment in $\Re^3$ with fixed position, tangent, and curvature at its end points. The quintic has 18 degrees of freedom; after the constraints are applied four degrees of freedom remain. Using two cubics we start with 24 degrees of freedom and after the constraints are applied 10 degrees of freedom remain. We must also consider the problem of joining these two segments with G^2 continuity, we take on considerable complexity while losing 7 more degrees of freedom, leaving three DOF with which to optimize the curve. In assessing the trade-off we must compare the loss of one degree of freedom, with the increased complexity, and with the loss of expressive power. Ultimately, the conclusion is subjective, little seems to be gained by lowering the order of the elements, the cost of minimization is approximately the same and the complexity is increased substantially.

Rational polynomials are very attractive as curve elements because they are capable of exactly representing portions of circles, helices, and torus curves. The prohibitive drawback of the rational form is the dramatic increase in complexity required to evaluate the MVC functional and its partial derivatives. This increase in complexity is due to the fact that, in contrast to integral polynomials, rational polynomials increase in order and complexity as derivatives are taken.

There are several possible representations for a quintic polynomial element (e.g. B-spline, Bézier). We chose the Hermite form because of the ease with which the geometric specifications can be mapped to the defining parameters of the Hermite segments. Also, this form is easily converted into other polynomial representations that are typically used in geometric modeling systems. Quintic Hermite curves are specified by the position of the endpoints and by the first two parametric derivatives at these locations. In vector notation this can be expressed as:

$$\vec{C}(u) = \begin{bmatrix} \vec{C}(0) \\ \vec{C}'(0) \\ \vec{C}''(0) \\ \vec{C}''(1) \\ \vec{C}'(1) \\ \vec{C}(1) \end{bmatrix}^{T} \cdot \begin{bmatrix} H_0(u) \\ H_1(u) \\ H_2(u) \\ H_3(u) \\ H_4(u) \\ H_5(u) \end{bmatrix} = \vec{C}(0)H_0(u) + \vec{C}'(0)H_1(u) + \ldots + \vec{C}(1)H_5(u)$$

where $H_i(u)$ are quintic blending functions. The computation of $H_i(u)$ is described in [59]. The mapping from the geometric to parametric description of the curve is carried out using the following equations

$$\vec{P}_i = \vec{p}_i$$

$$\vec{P}'_i = m_i \hat{t}_i \qquad \textbf{(4.7)}$$

$$\vec{P}''_i = m_i^2 \vec{\kappa}_i + \alpha_i m_i \hat{t}_i \qquad \textbf{(4.8)}$$

where $\vec{p}_i$, $\hat{t}_i$, $\vec{\kappa}_i$ are the position, tangent direction, and curvature vector at one end of the curve; m_i is the first derivative magnitude and α_i completes the relationship between curvature and the second derivative. During the minimization the scalar m_i must be constrained to be positive; this is because if m_i became negative then P'_i would reverse direction. We impose this constraint by using m_i^2 rather than m_i in equations (4.7) and (4.8):

$$\vec{P}_i = \vec{p}_i$$

$$\vec{P}'_i = m_i^2 \hat{t}_i$$

$$\vec{P}''_i = m_i^4 \vec{\kappa}_i + \alpha_i m_i^2 \hat{t}_i .$$

A curve is made up of a sequence of vertices connected by quintic elements. G^2 curves are pieced together from these elements by sharing geometric specifications at the common points. Data structures associated with the vertices hold the point, tangent, and curvature information, while data structures associated with the elements hold the parameters $\alpha_{i,0}$, $\alpha_{i,1}$ and $m_{i,0}$, $m_{i,1}$ (Fig. 4.4). Each element is defined by three data structures, a pair of structures associated with the vertices at the element's endpoints and one structure associated with the element itself. By distributing the curve/element specification in this way, adjacent elements share vertex structures and are guaranteed to meet with curvature continuity. Note that discontinuities can be introduced by giving adjacent elements independent geometric specifications.

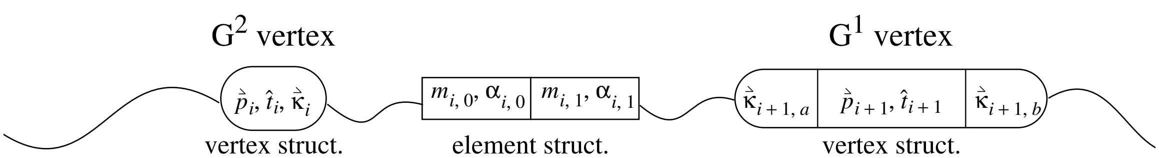


Figure 4.4. Schematic View of Curve Representation.
On the left is a G^2 joint where incident curves share the full geometric specification. On the right is a G^1 joint where incident curves have independent curvatures.

4.6 Multi-element Segments

As we have described thus far, a single quintic Hermite segment or *element* is placed between every two sets of constraints. Because of the limited descriptive power of polynomial elements, a single element can only approximate the ideal minimum variation curve. To improve the approximation, multiple elements can be inserted between constraints. In practice, a single element per constraint pair is normally sufficient. Depending on the goal of the application, it may not be important that the minimum variation curve is accurately approximated, only that its desirable curvature properties be present. As an example, the *Wicket* in Figure 4.2 on page 71 is reexamined in the context of using multiple elements per pair of constraints. The MVC's characteristics for these constraints are simply those of a semicircle; MVC functional is 0.0, the arc length is π, and the curvature κ is 1.0. The *Wicket* was calculated using one and two elements. The one element approximation's maximum and minimum curvatures were 1.00013 and 0.99986 respectively. The two element approximation's extremal curvatures were 1.0000079 and 0.9999923. The arc length of the two element approximation was within 10^{-6} of π. We see that, in this example a single element produces a good approximation to the MVC and that a two element solution is extremely close to the theoretical minimum variation curve.

4.7 Parametric Functionals

The MVC curvature functional (4.2) is defined in terms of an integral of a function over arc length. To evaluate this functional and its gradient in the context of the parametric piece-wise polynomial curves described in section 4.5, the functional must be converted to a parametric polynomial form and evaluated in a piecewise fashion. The value of the functional for the curve as a whole is computed as the sum of the values of the functional for each element. In the first conversion step the arc length based definition

$$\int_0^l \frac{d\vec{\kappa}}{ds}^2 ds$$

is changed to an integral of a function of the curve $\vec{C}(t)$ parameterized by t of the form

$$\int_\alpha^\beta f(\vec{C}(t))dt.$$

The bounds, α, β, of the integral are set to 0 and 1, since the Hermite representation is parameterized with t varying from 0 to 1. The differential with respect to s is converted to a differential in t. Since

$$\frac{dt}{ds} = \frac{1}{\left\|\vec{C}'(t)\right\|} \text{ then } ds = \left\|\vec{C}'(t)\right\| dt$$

where

$$\left\|\vec{C}'(t)\right\| = (\vec{C}'(t) \cdot \vec{C}'(t))^{1/2}.$$

Next, the derivative of curvature with respect to arc length $\frac{d\vec{\kappa}}{ds}$ transforms to: $\frac{d\vec{\kappa}}{dt}\frac{dt}{ds}$. These two steps yield:

$$\int_0^l \frac{d\vec{\kappa}}{ds}^2 ds = \int_0^1 \left(\frac{d\kappa}{dt}\frac{dt}{ds}\right)^2 \left\|\vec{C}'(t)\right\| dt \qquad \int_0^l \frac{d\vec{\kappa}}{ds}^2 ds = \int_0^1 \frac{d\vec{\kappa}}{dt}^2 \frac{1}{\left\|\vec{C}'(t)\right\|} dt. \qquad \textbf{(4.9)}$$

Lastly we find the expression for $\frac{d\vec{\kappa}}{dt}$ in terms of the parametric derivatives of the curve, $\vec{C}(t)$. The expression for curvature is

$$\vec{\kappa}(t) = \frac{\vec{C}'(t) \times \vec{C}''(t)}{\left\| \vec{C}'(t) \right\|^3}$$

Taking the derivative with respect to t yields

$$\frac{d\kappa}{dt} = \frac{v\,du - u\,dv}{v^2}$$

where

$$u = \vec{C}'(t) \times \vec{C}''(t), \qquad du = \vec{C}'(t) \times \vec{C}'''(t), \qquad \text{n.b. } \vec{C}''(t) \times \vec{C}''(t) = 0,$$

$$v = \left\| \vec{C}'(t) \right\|^3 = (\vec{C}'(t) \cdot \vec{C}'(t))^{3/2}$$

$$dv = 3\left\| \vec{C}'(t) \right\| (\vec{C}'(t) \cdot \vec{C}''(t))$$

4.8 Computing Partial Derivatives

The optimization process we use requires that we compute the partial derivatives of the functional. Since the derivative of an integral is equal to the integral of the derivative, we must find the partial derivatives of the integrand of (4.9). We discuss two methods that can be used to compute these partial derivatives: 1) analytical or 2) *central differencing*. All the examples shown in section 4.11 were computed using analytically computed partial derivatives. The primary advantage of using central differencing is the ease with which it may be implemented. Briefly, derivatives are approximated by computing the value of the subject function while varying the value of the function parameter; Conte and de Boor [39] is a good reference covering central differencing. The primary advantage of computing the partial derivatives analytically is the greater numerical precision with which they can be calculated. The added precision increases the rate of convergence of the optimization.

It is advantageous to make extensive use of the chain rule and to compute the partial derivatives using a number of steps. Taking this approach, common subexpressions are revealed. For example the first few steps in computing the partial derivative of the integrand of (4.9) are

$$\frac{\partial\left(\frac{d\vec{\kappa}}{dt}^2 \frac{1}{\left\|\vec{C}'(t)\right\|}\right)}{\partial\alpha} = \frac{\partial\left(\frac{d\vec{\kappa}}{dt}^2\right)}{\partial\alpha}\frac{1}{\left\|\vec{C}'(t)\right\|} + \left(\frac{d\vec{\kappa}}{dt}\right)^2 \frac{\partial\left(\frac{1}{\left\|\vec{C}'(t)\right\|}\right)}{\partial\alpha}$$

$$\frac{\partial\left(\frac{d\vec{\kappa}}{dt}^2\right)}{\partial\alpha} = 2\frac{d\vec{\kappa}}{dt}\frac{\partial\left(\frac{d\vec{\kappa}}{dt}\right)}{\partial\alpha}$$

$$\frac{\partial\left(\frac{1}{\left\|\vec{C}'(t)\right\|}\right)}{\partial\alpha} = \frac{-\frac{\partial\left\|\vec{C}'(t)\right\|}{\partial\alpha}}{\vec{C}'(t)\cdot\vec{C}'(t)}$$

$$\ldots$$

The process of computing the derivatives appearing in subexpressions continues until the partial derivative of (4.9) is fully evaluated. The resulting expression tree is too large to reproduce here. The leaves of this expression tree are the partial derivatives of $\vec{C}(t), \vec{C}'(t), \vec{C}''(t)$ with respect to α.

The computation of partial derivatives is accomplished by implementing a suite of functions which reference and assign values in a block of global variables. Each step in the chain rule corresponds to a function; the result of each function call is stored in a global variable for use by subsequent function calls. To guarantee that the variables that a function requires for evaluation have been initialized, an analysis of the cross references of the functions and global variables is used to produce an appropriate ordering of the functions used to evaluate each partial derivative.

4.8.1 Numerical Integration

During the optimization process, the values of the MVC functional and its partial derivatives with respect to the free parameters are computed. Since the value of the MVC integral cannot be computed in closed form, we sample the function over a domain to approximate the integral;

$$\sum_i w_i f(t_i),\ t_i \in [\alpha, \beta] \approx \int_\alpha^\beta f(t)\,dt.$$

Gauss-Legendre quadrature is used to compute the (curvature) integrals. This method converges quickly as the number of samples or integration points is increased [172].

4.8.2 Gradient Descent

Cast as a multi-dimensional optimization problem, the curve is represented by a point in $\Re^n$ corresponding to its n degrees of freedom. The MVC functional undergoing minimization is expressed as $f(x), x \in \Re^n$. Standard optimization techniques are used to minimize $f(z)$. Starting from a heuristically established point in $\Re^n$, Polak-Ribiere conjugate gradient descent [172] is used to traverse the space while reducing the objective function and ultimately arriving at a minimum. This descent method uses a weighted average of gradients from past iterations,

$$\tilde{\nabla} = \sum \nabla f(x_i) w_i,$$

to determine a direction for movement $\tilde{\nabla}$. Once $\tilde{\nabla}$ is computed, a one-dimensional minimization is performed to find the minimum of $f(x - \alpha\tilde{\nabla})$ over α. First the line minimization brackets a minimum, finding three values of α such that $f(x - \alpha_2\tilde{\nabla}) < f(x - \alpha_1\tilde{\nabla})$, $f(x - \alpha_2\tilde{\nabla}) < f(x - \alpha_3\tilde{\nabla})$ and $\alpha_1 < \alpha_2 < \alpha_3$. Then it uses a parabolic fit to calculate a new value of α, and eliminates either α_1 or α_3 while maintaining the bracketing relationship. This second step is repeated until a minimum is reached when the α_i converge on each other, (Fig. 4.5).

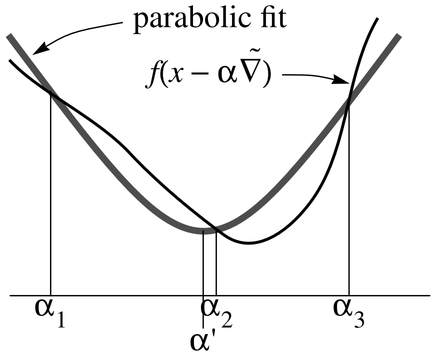


Figure 4.5. Parabolic Line Minimization.

Line minimization using a parabolic fit to find a new α, here $\alpha', \alpha_2, \alpha_3$ are used as the next bracketing triple.

4.9 Initialization

The descent scheme described in section 4.8 starts with an initial curve and iteratively refines that curve until the optimal curve is achieved. In this section we discuss the problem of finding a suitable initial curve. In terms of optimization, the problem is to find an initial point in the "valley" where the minimum point (corresponding to the stable equilibrium of shortest length) lies at the bottom (see section 4.10). The optimization requires that initial values be provided for any parameters not explicitly set; m_{i0}, m_{i1}, α_{i0}, α_{i1}, $\hat{t}_i$, and $\vec{\kappa}_i$ in equations (4.7), (4.8), and Figure 4.4. Several researchers have studied the problem of finding an interpolating curve for an ordered set of points (3.1.3).

The rate of convergence of the minimization procedure depends strongly on the initial curve and, as a result, so does the speed with which solutions may be found. We use our knowledge of the nature of the MVC functional to choose an initial curve shape that is close to the solution. The formulas that we use for the initial guess are taken from [28]. The values of m_{i0} and m_{i1} are related to the arc length of the curve and are initialized to the chord length $\|\vec{p}_i - \vec{p}_{i+1}\|$. The values of α_{i0} and α_{i1} specify the nature of the parameterization at the endpoints; setting them to 0 causes the curve to be arc length parameterized at its endpoints. The tangent directions are set to the average of the incident chord directions weighted inversely in proportion to their length (Fig. 4.6):

$$\grave{t}_i = \frac{\vec{p}_{i-1} - \vec{p}_i}{\|\vec{p}_{i-1} - \vec{p}_i\|^2} + \frac{\vec{p}_i - \vec{p}_{i+1}}{\|\vec{p}_i - \vec{p}_{i+1}\|^2} \qquad \hat{t}_i = \frac{\grave{t}_i}{\|\grave{t}_i\|}. \tag{4.10}$$

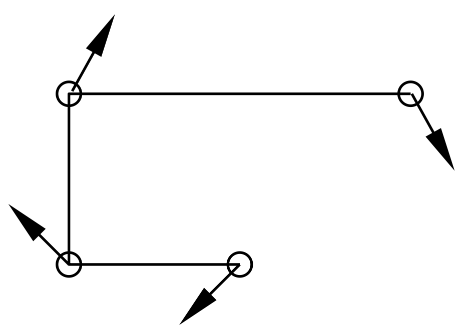


Figure 4.6. Tangent Initialization.
The tangent is computed as the inversely weighted average of incident chords, see eq. (4.10).

A number of schemes have been put forth for heuristically assigning tangent directions. These include a strict average of chord directions, and a weighting in proportion to the length of the chord. If we consider the nature of the MVC functional, it is clear that (4.10) is the appropriate formula. The key argument is based on consistency (MVC *are* consistent.) Given an existing curve passing through two points, if a third point is added that lies on the curve between the first two and is close to first, then any weighting of incident chords other than (4.10) will yield a poor tangent estimate. In fact, if the curve is circular then for the introduction of any intervening point equation (4.10) yields the tangent to the circle at that point; precisely the desired tangent.

The magnitudes of the curvature vectors $\vec{\kappa}_i$ are set by computing the radius of the circle that interpolates the associated point and its two neighbors (Fig. 4.7). The direction of the curvature vector is perpendicular to the tangent and is in the plane of the three points defining the circle

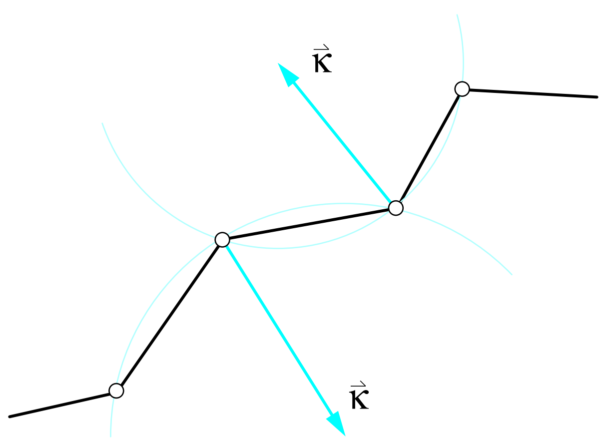


Figure 4.7. Curvature Initialization.
Curvature vectors are initialized by fitting circles through adjacent points.

In the case where the curve is not closed and the elements at the ends of the curve have endpoints that do not have two neighbors, the curvature is set equal to the curvature to which the single neighbor was initialized. The tangent direction is set to the tangent direction at the other endpoint reflected about the perpendicular plane through the midpoint of the intervening chord (Fig. 4.8).

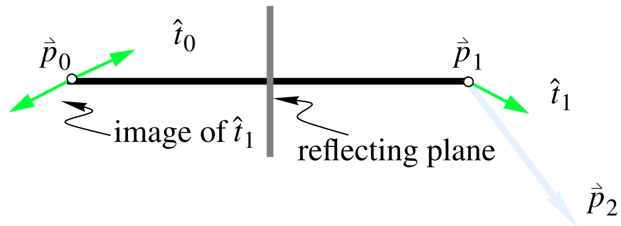


Figure 4.8. End Point Tangent Construction.
Construction of tangent direction at the open end point of a curve.

Our heuristic technique for selecting a starting point has proven to be efficient and robust. If difficulties are encountered, the continuum method described in section 4.10 may be employed, which starts with a penalty for arc length and then gradually relaxes that penalty.

4.10 Existence, Uniqueness, and Sensitivity

Because the MEC has been much more widely studied (since the 17th century), we discuss results from these studies and relate them to the relatively new MVC. In [104] Jerome discusses the necessary and sufficient conditions for the existence of an MEC. To the authors' knowledge there has been no study of the uniqueness of MVC.

In [65] Fisher and Jerome discuss the stable and unstable equilibrium of an MEC. An MEC is said to be at stable equilibrium if it holds its shape in the absence of an arc length constraint. In most practical applications we seek the stable equilibrium solution that has the shortest length. This suggests the use of the continuum method, the approach of computing MVC by initially placing the curve under tension (by penalizing arc length) and gradually relaxing the tension until stable equilibrium is achieved and no tension remains. Figure 4.9 illustrates two MVC curves for the same set of geometric constraints. In addition to the positional constraints, the curvature at the central point is fixed. In comparing the two solutions, note that while the solution without the loop is shorter, it has a larger MVC functional value.

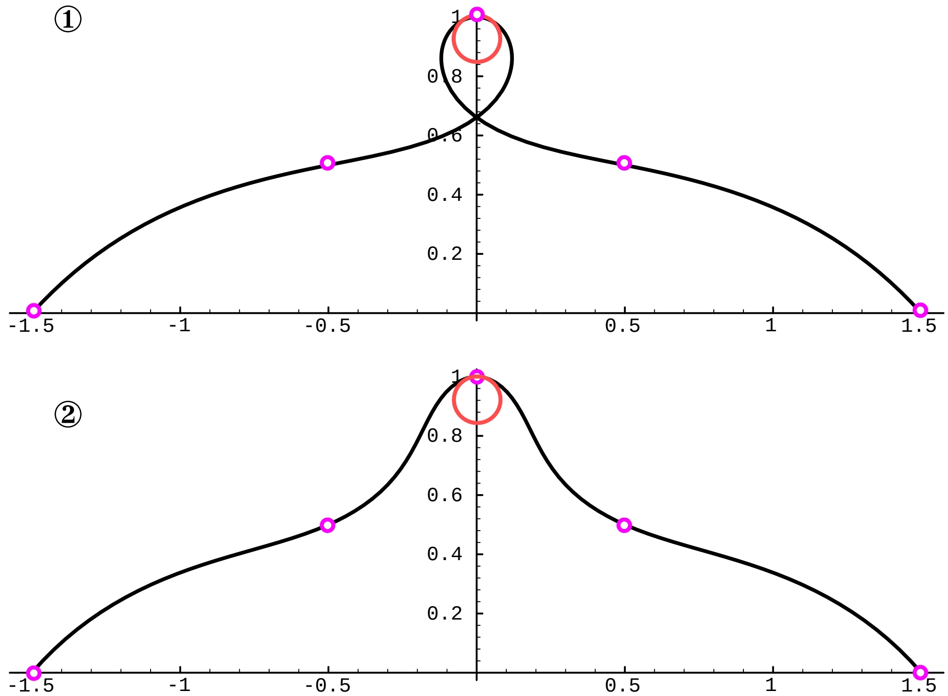


Figure 4.9. Multiple MVC Curves from One Specification.

① the MVC resulting from an initial curve containing a loop. ② the MVC resulting from an initial curve without a loop. Note that while curve ② is probably the desired curve, curve ① has a lower MVC functional value. The curves are specified by five positional constraints and a single curvature constraint at the center of the curve. Note how the MVC distributes the high curvature at its midpoint.

4.11 Results: Test Cases, Curve Quality, and Applications

In order to evaluate the utility of MVC, we discuss the solution to a simple problem, and compare MVC with other curve representations; we then present a few sample applications. Comparisons are done using a number of different techniques: by drawing curvature and torsion plots, and by examining curvature profiles [98]. Curvature profiles are generated by drawing the reciprocal of the evolute of a curve; this is traced out by the curve normals scaled by the curvature of the curve.

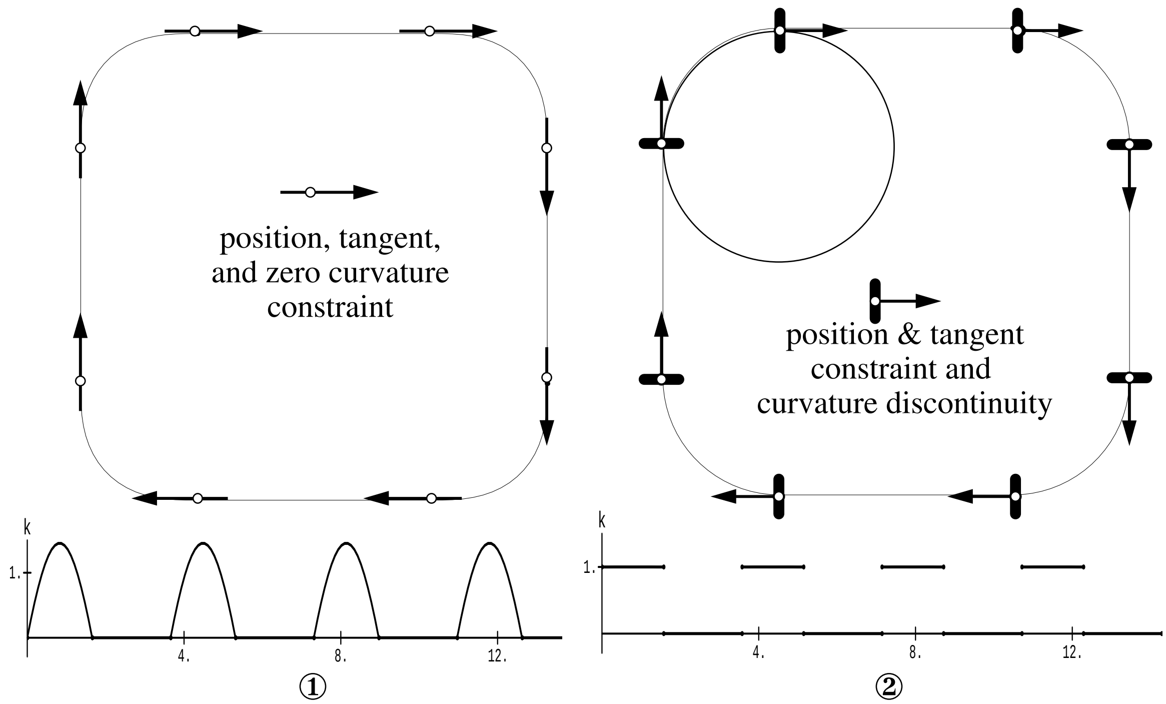


Figure 4.10. Blending the Corners of a Box.

① Curvature continuous blends specified with position, tangent and zero curvature constraints. ② Circular blends specified with position and tangent constraints, and curvature discontinuities.

4.11.1 A Sample Problem: Corner Blending

In this example eight positional constraints are set symmetrically near the corners of a box, our goal is to round the corners of the box. Two possible solutions to the blending problem are presented. Figure 4.10① illustrates the case where tangent and curvature constraints are specified at the eight points. These constraints result in a continuous curvature plot, with value zero along the sides of the box, and varying from zero to a peak of around 1.5 at the maximum. The second solution, (Fig. 4.10②) inserts curvature discontinuities at the blend boundaries, which allows the MVC to form circular blends at the corners of the square. Comparing these two blend alternatives, the first is curvature continuous at the expense of a slightly higher maximum curvature, and the second is tangent continuous and circular.

4.11.2 A Comparison of MVC, MEC and Natural Splines

Figure 4.11 highlights the differences in fairness exhibited by a natural spline, an MEC, and an MVC constrained to pass through a given set of positional constraints [211]. Though the curves shown in the top of the figure are nearly indistinguishable on paper, the curvature plots emphasize the differences between these curve generation techniques. The natural spline has a "spiky" curvature plot with peaks at several of the constraint points, and zero curvature at the endpoints. The MEC exhibits a considerably smoother curvature plot with slope discontinuities at the interpolated points, and zero curvature at the endpoints. Finally, the MVC has a very smooth curvature plot, free of sharp peaks and corners, and constant curvature at the endpoints. Note also that the MVC has a considerably lower maximum curvature.

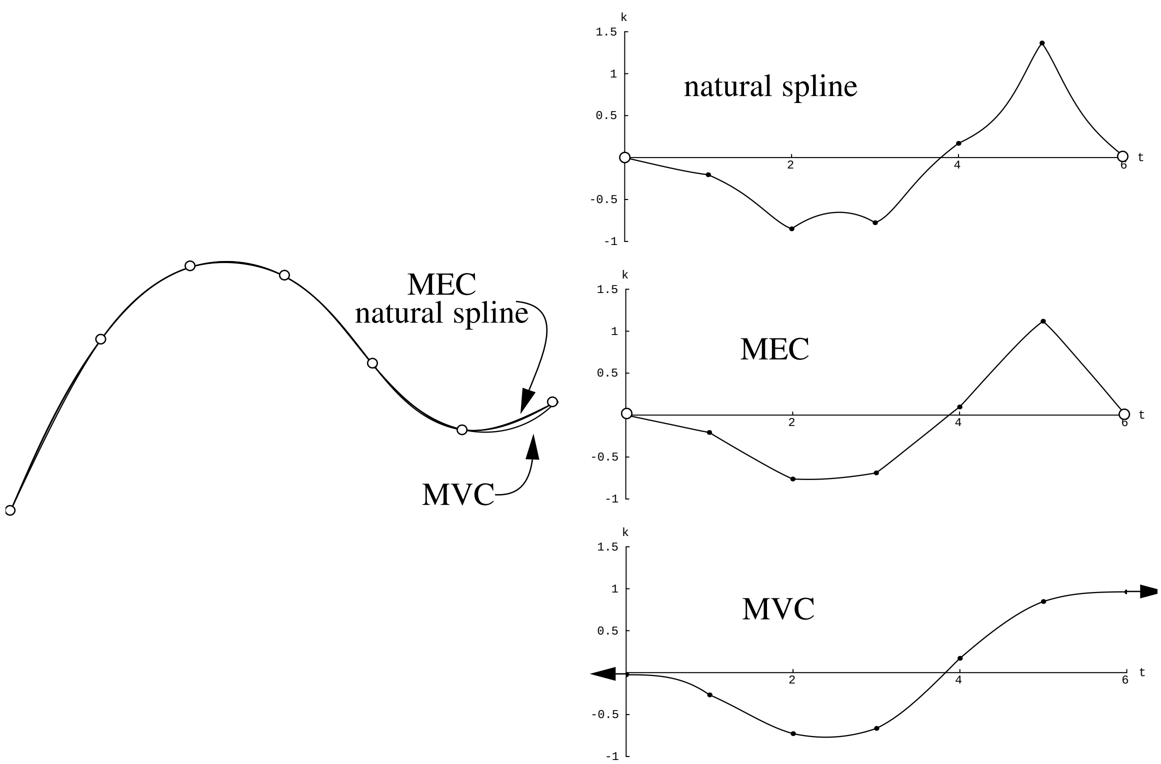


Figure 4.11. MVC vs. MEC and Natural Splines.

A comparison of natural splines, MEC, and MVC. Note that the MVC has a much smoother curvature distribution and, has constant curvature at its endpoints rather than zero curvature.

4.11.3 Scale-Independent MVC

First we compare the curvature plots of the MVC and SI-MVC for the data set presented in Figure 4.11 (Fig. 4.12). We see that the SI-MVC has a smooth curvature plot, nearly indistinguishable from the standard MVC. It is only in cases where curvature variation is high that there is a noticeable difference between MVC and SI-MVC. In Figure 4.13 we

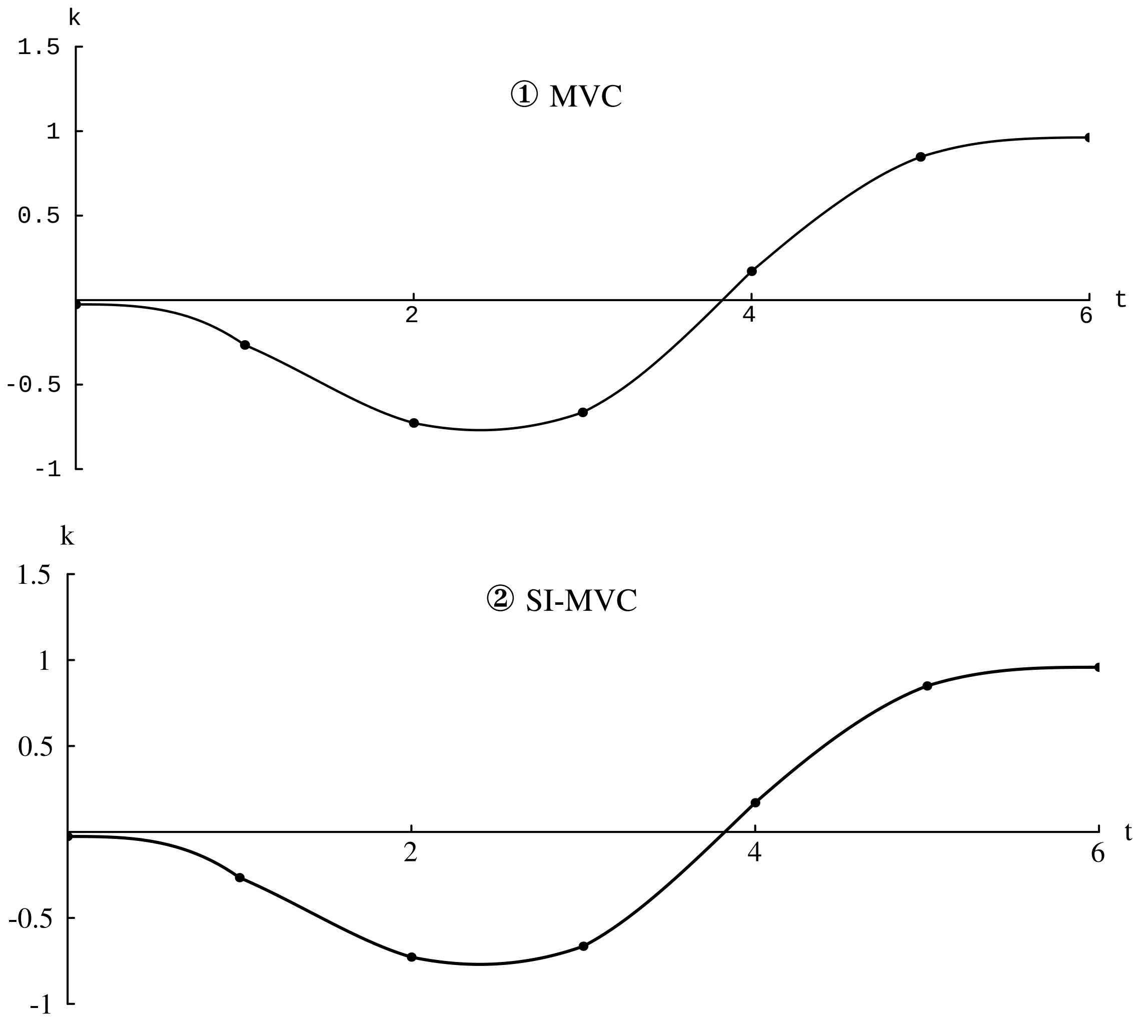


Figure 4.12. Curvature Plot—MVC and Scale-Independent MVC

There is virtually no difference between MVC ① and standard SI-MVC ② for problems with low curvature variation. These curvature plots result from the Woodford data set, also shown in Figure 4.11.

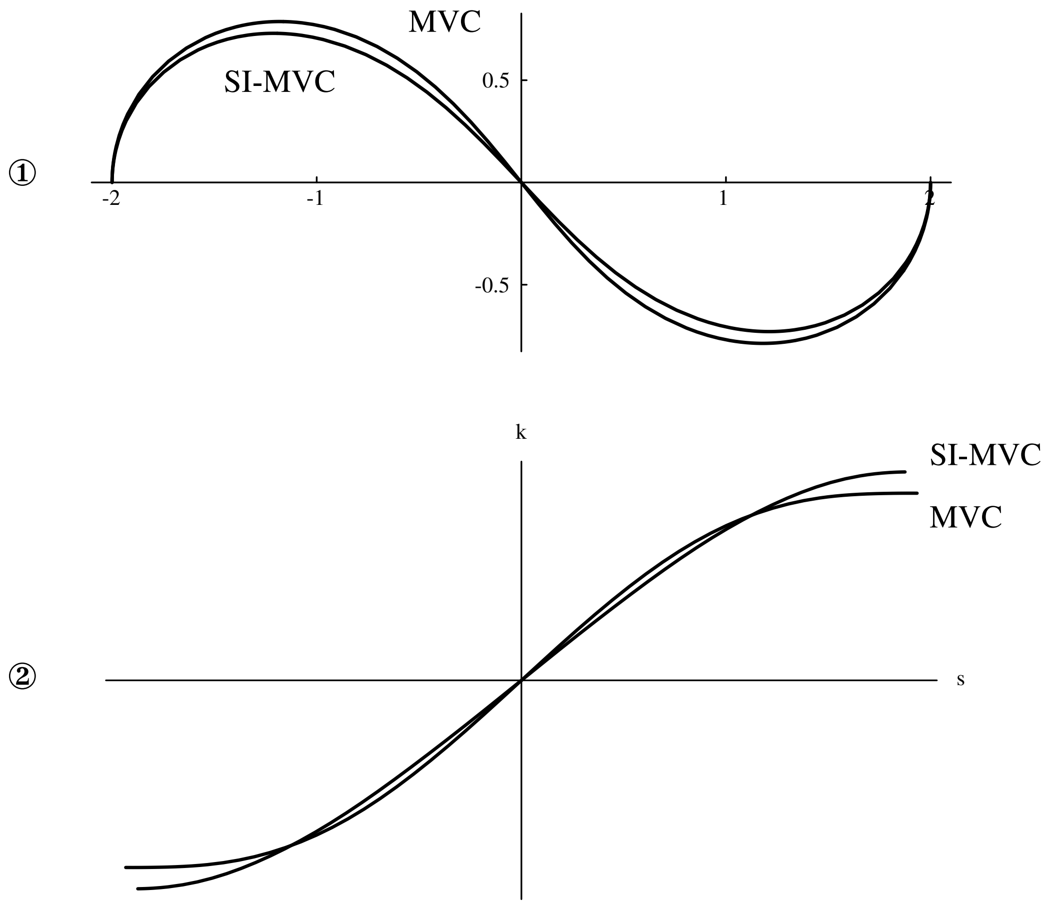


Figure 4.13. Planar S-shaped Curves, MVC vs. SI-MVC

These curves are specified by position and tangent constraints at their end points. The SI-MVC is somewhat tighter, shorter in length and its curvature varies more linearly. The MVC has constant curvature at its end points. ① is a plot of the curves, ② are the corresponding curvature plots.

see *S*-shaped MVC and SI-MVC, specified by constraints on position and tangent direction at the curve's end points. The standard MVC is somewhat longer and fuller in appearance. If the tangents specifying this curve are rotated further, so that the curve must bend even more, the standard MVC jumps to infinity since it can reduce the value of its functional by letting the curve expand. The SI-MVC, however remains finite even for the case of antipodal tangent constraints (Fig. 4.14).

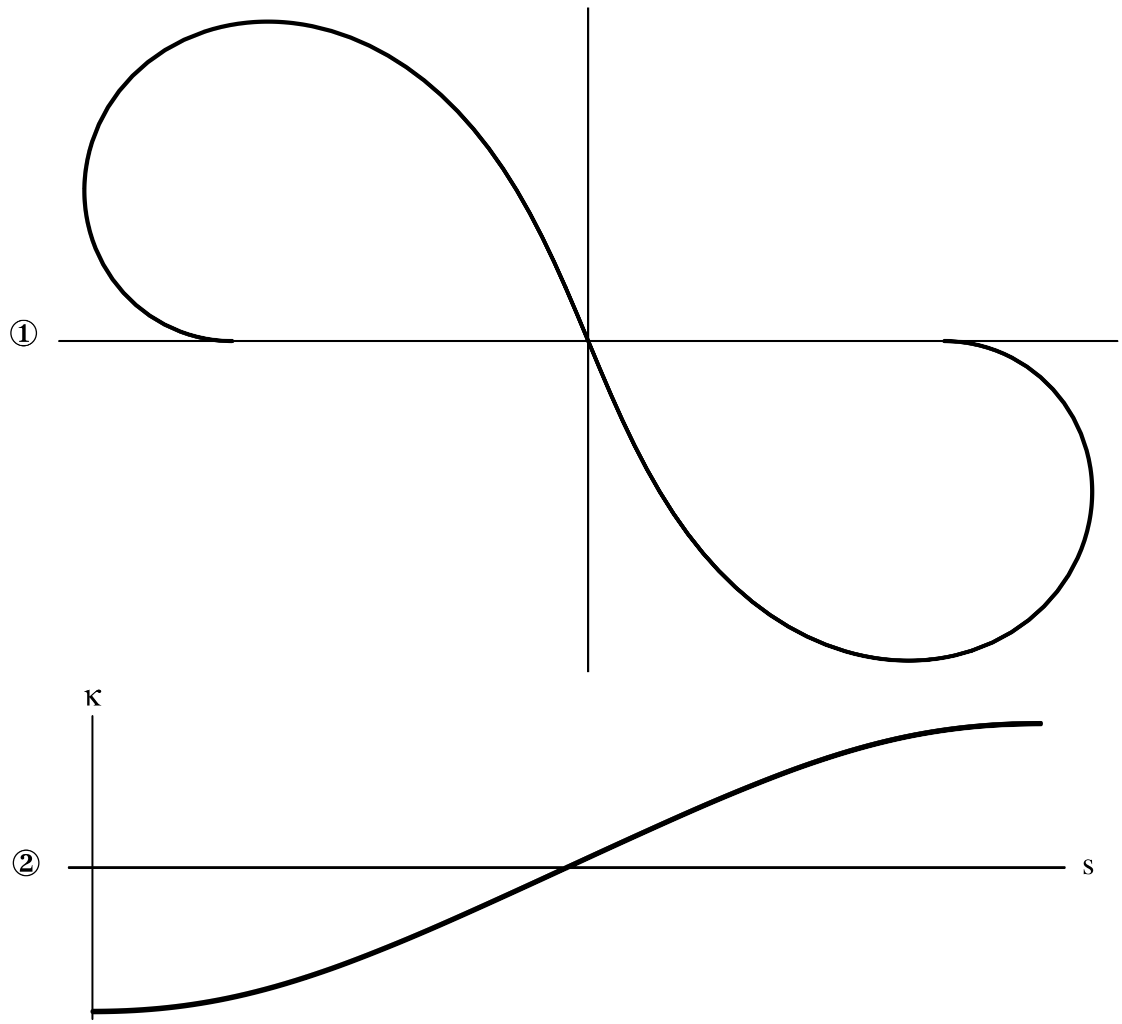


Figure 4.14. A Curve from Antipodal Tangent Constraints
① the SI-MVC resulting from antipodal tangent constraints. The MVC expands to infinity for this constraint set. ② the curvature plot of the SI-MVC, curvature vs. arc length.

As an example to demonstrate scale invariance we compute the shape of a completely unconstrained figure-8. We initialize the curve to the rough shape of a figure-8 and then optimize its shape using the SI-MVC functional without any constraints whatsoever. In Figure 4.15 we provide the curve and curvature plot; the curve forms a lemniscate-like curve. Additional experiments reveal that, if an additional loop is added to the initial figure-8, the extra loop forms an osculating circle at the tip of the lobe, where the

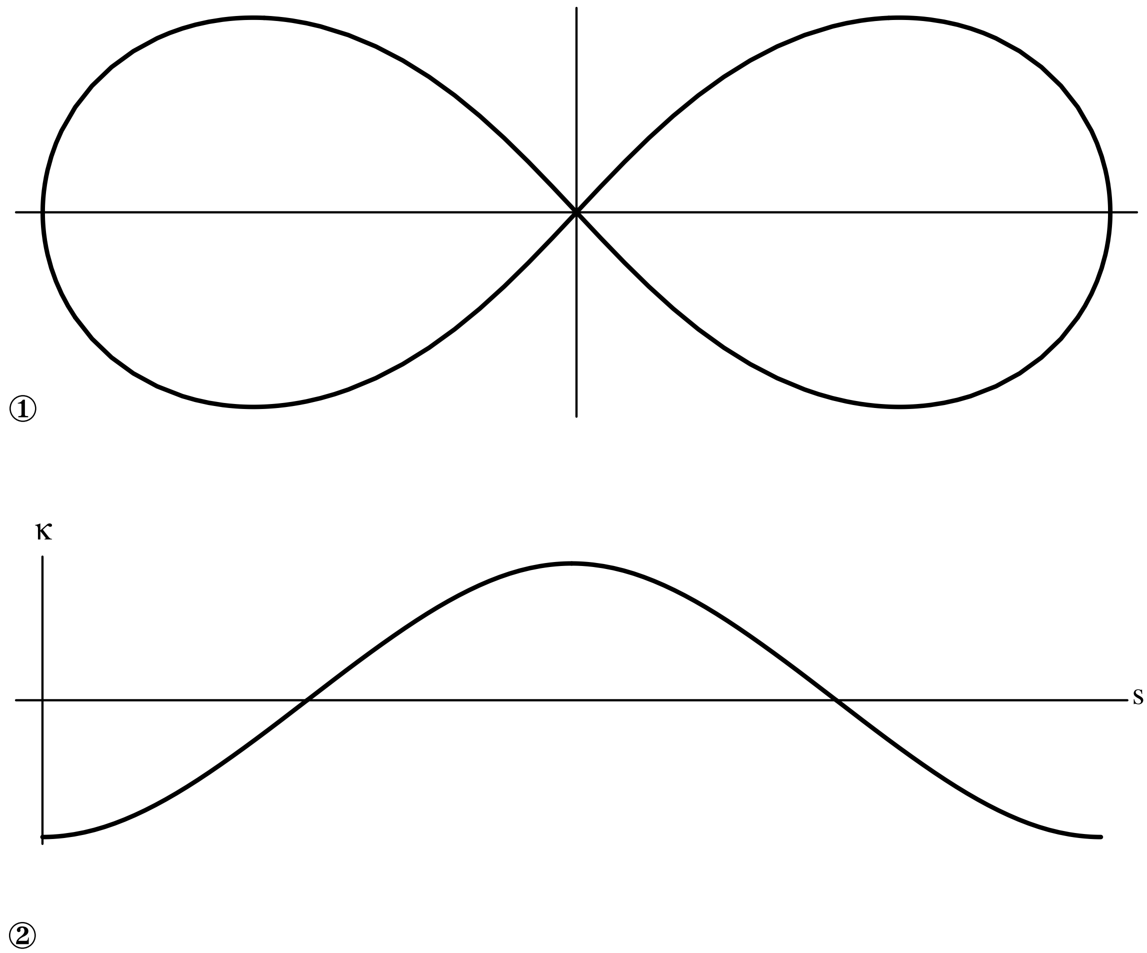


Figure 4.15. An SI-MVC Figure-8.
① an unconstrained figure-8. ② the curvature plot, κ vs. arc length.

derivative of curvature is zero. If we had attempted the same optimizations using the standard MVC functional, the optimization procedure would have expanded the curve's scale to infinity reducing curvature variation.

The primary drawback of SI-MVC is that their rate of convergence is much slower than that of the standard MVC. For example, the MVC generated from the Woodford data set converged in 137 iterations and the SI-MVC took a total of 3921 iterations. Future research will investigate the cause of the poor convergence rate of the SI-MVC.

4.11.4 MVC vs. MEC Space Curves

In this example we examine the characteristics of the MVC and MEC *space curves.* Figure 4.16 is a schematic view of the constraints used in this example. Figure 4.17 shows the MVC to be smoother and of lower maximum curvature. Note also that the MVC, unlike the MEC is torsion continuous. As alternative representation we display these two curves using curvature profiles (Fig. 4.18).

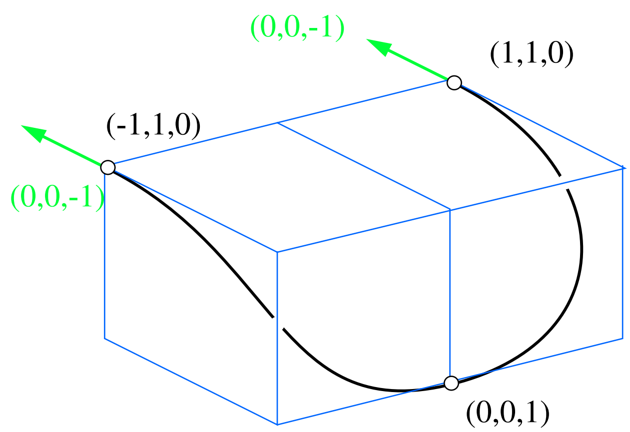


Figure 4.16. A simple space curve for comparing MVC with MEC.

As a second example applied to non-planar data we examine the results of computing the MVC and MEC fit to the helical data of Jörg Peters [157]. Here we see the MEC has discontinuities of curvature and zero curvature at the end points of the curve. The MVC exhibits a very uniform distribution of curvature. Figure 4.19 illustrates the starting shape for the optimization, Figure 4.20 shows the MEC, and Figure 4.21 the MVC.

4.11.5 Coving Design

In this example, we describe an actual application of MVC to a real-world architectural design problem. For a room with a 6" heating duct running along where the ceiling meets the wall, design a coving to cover the duct that meets the wall and ceiling smoothly. The coving may be no more than 12" high and 20" deep (Fig. 4.22). The coving was designed by imposing the position and tangent constraints implied by the problem parameters, and

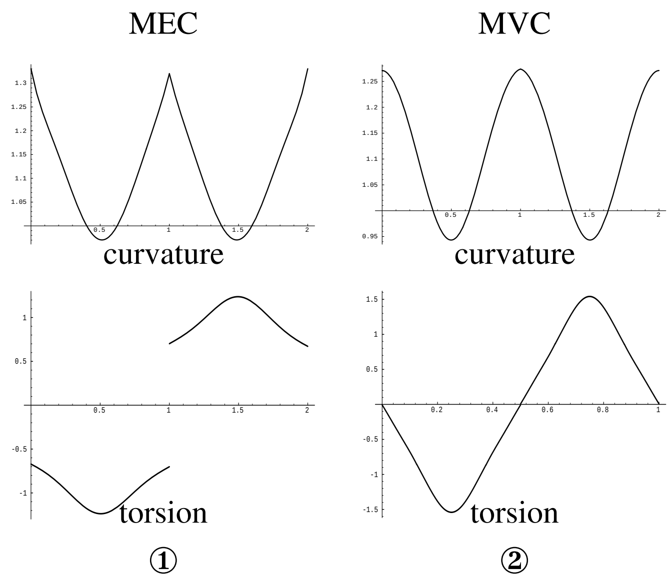


Figure 4.17. Space Curve Curvature and Torsion Plots.

Curvature and torsion plots corresponding to the constraint set shown in Fig. 4.16. The MVC has smoother curvature and torsion and is torsion continuous.

interactively adjusting the curvature of the coving where it met the wall. The curvature was increased until the coving cleared the duct by a reasonable amount. The coving was then cut out of high density foam by an NC-controlled cutter taking a spline path as input.

4.12 Efficiency

In this section we present wall clock timings taken on a 33MHz IRIS Indigo. Table 1 lists timings for each of the examples in section 4.11, and also for the Wicket from Figure 4.2. Each optimization was carried out for both the MEC and MVC functionals and was terminated when the infinity norm of the gradient was less than 0.0005. For these examples, the MVC takes longer to compute, except when higher order constraints are applied, then the MEC tends to take longer. However, in all cases the strain energy of the MVC was closer to the energy of the MEC than the curvature variation of the MEC was to the variation of the MVC. In other words, an MVC approximates an MEC better than an MEC approximates an MVC.

We also provide plots of the log of the MVC functional value vs. number of iterations of the optimization procedure in Figure 4.23 and Figure 4.24. The functional values are normalized by subtracting the minimum value and adding one; this ensures that the

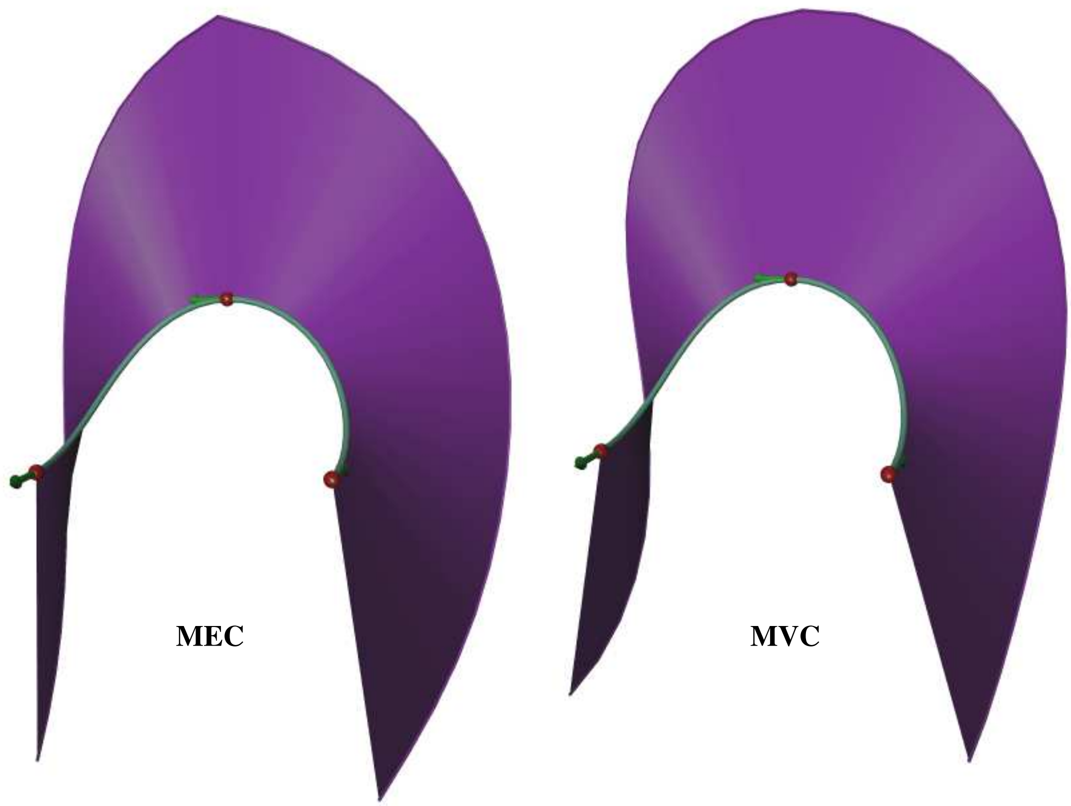


Figure 4.18. Curvature Profiles of the Simple Space Curve (Fig. 4.16).
Note the slope discontinuity in the curvature of the MEC, this corresponds to the torsion discontinuity.

minimum plotted log value will always be zero. The data plotted are discrete in nature and are interconnected to make trends more obvious. From these examples it is clear that the majority of the reduction in the MVC functional occurs very quickly. And for this reason it is practical to build an interactive editor using MVC. Also because of this low computation time, MVCs have become an attractive alternative to traditional methods for curve design.

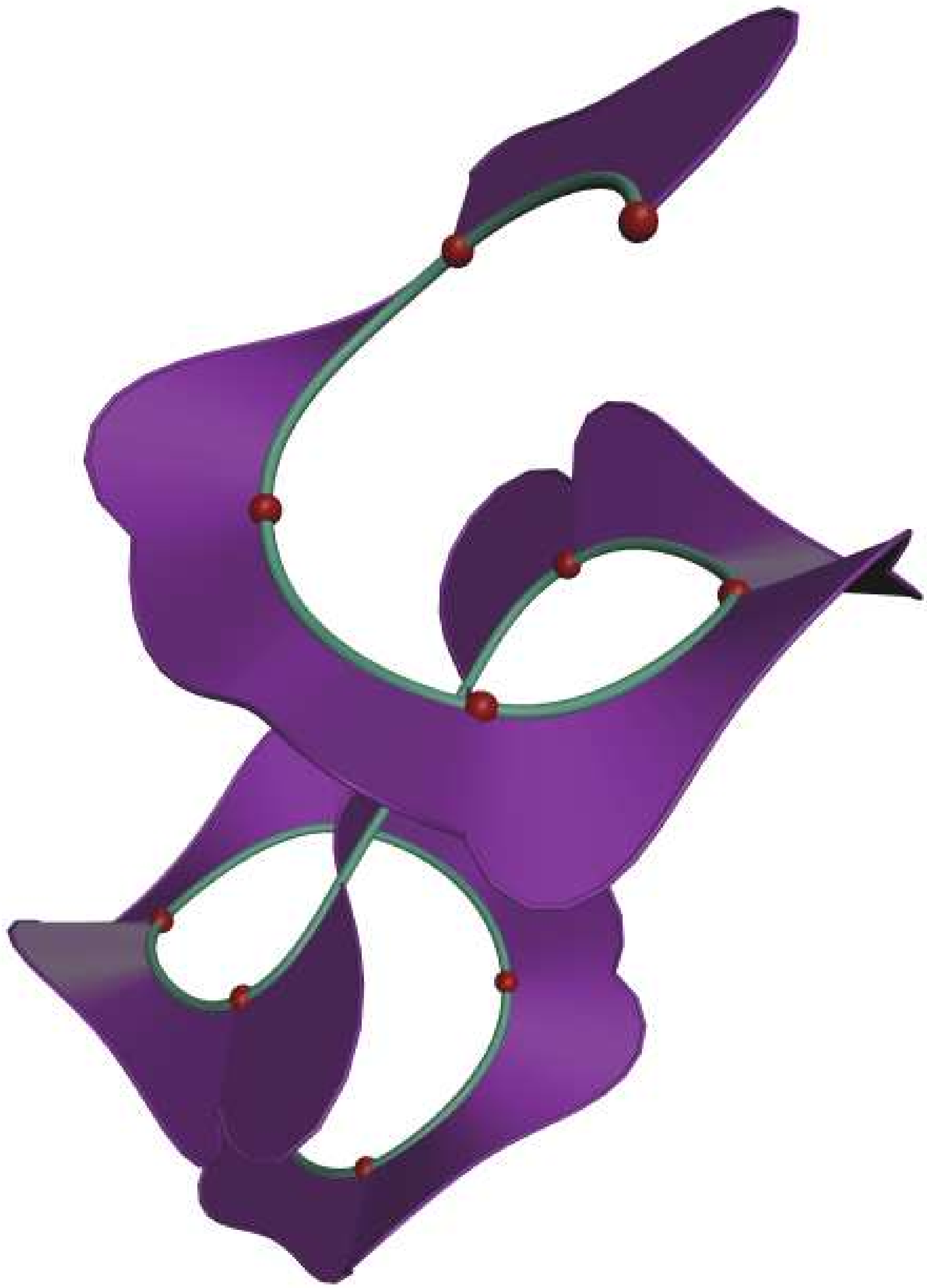


Figure 4.19. Jörg Peters' Helical Data — The Initial Curve
The heuristic initial guess produces a curve with uneven curvature and torsion.

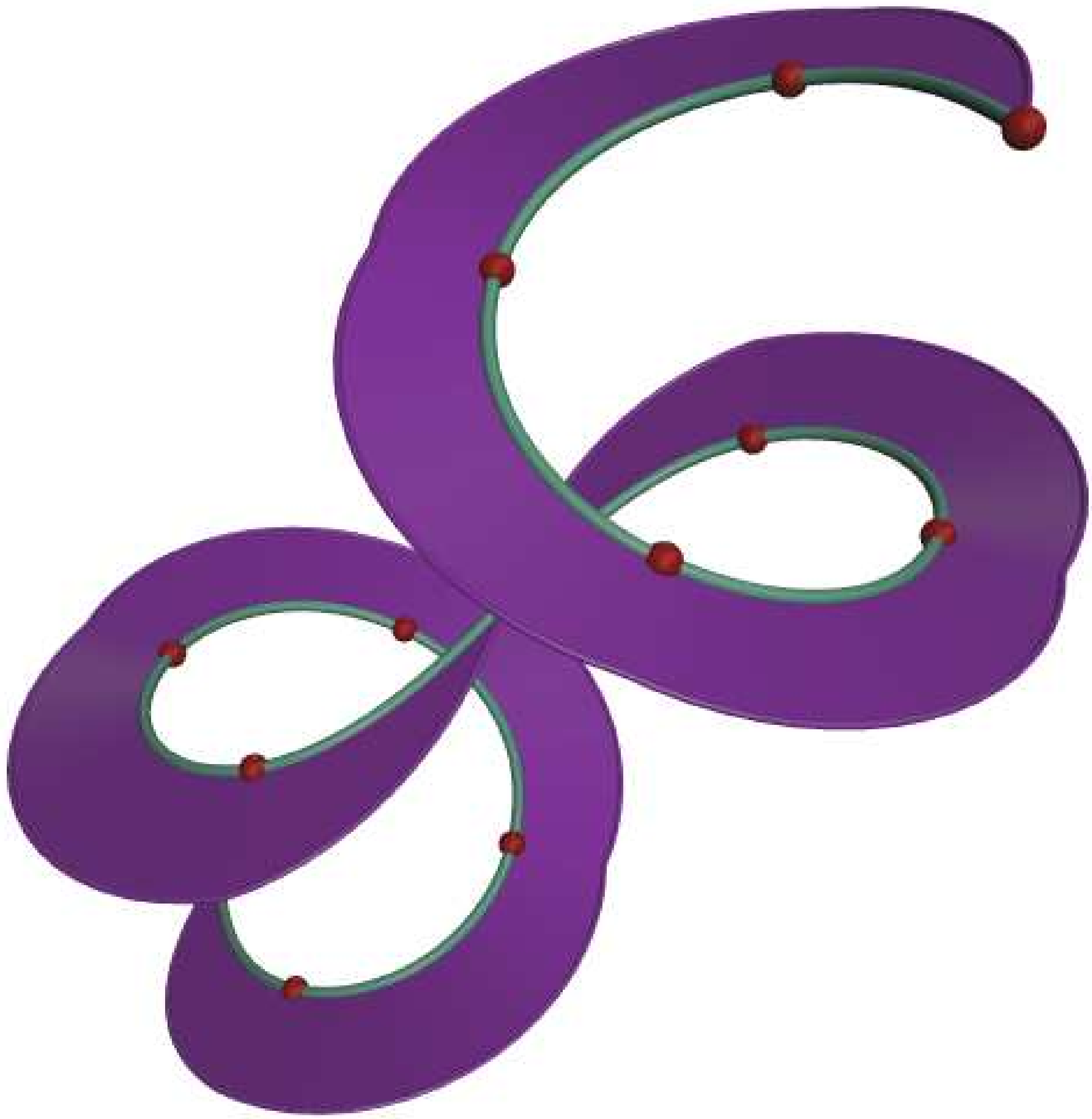


Figure 4.20. Jörg Peters' Helix — The MEC Curve

The MEC curve has zero curvature at its end points and slight discontinuities of the derivative of curvature at each interpolation point.

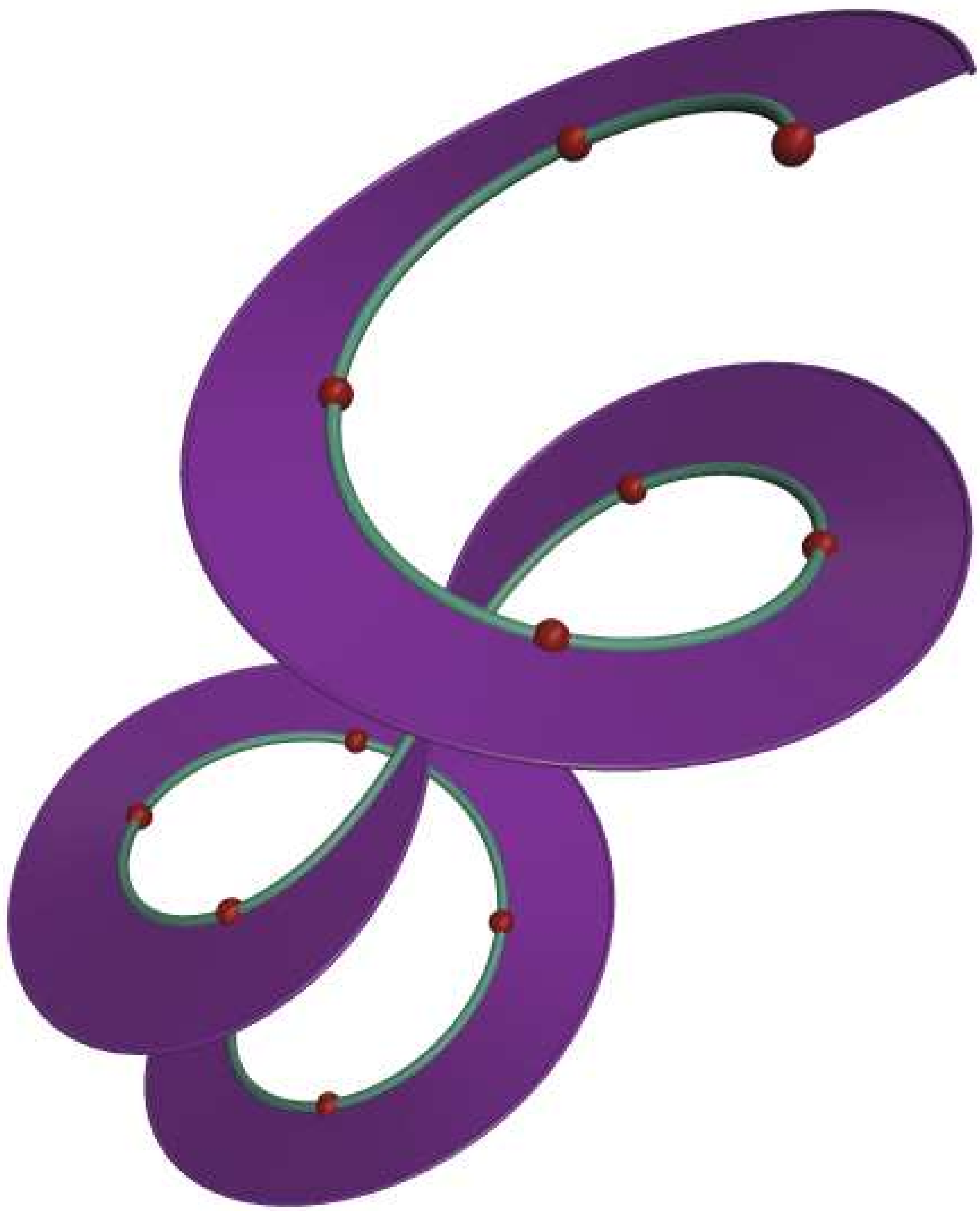


Figure 4.21. Jörg Peters' Helix — The MVC Curve
The MVC curve has curvature and torsion of nearly constant magnitude.

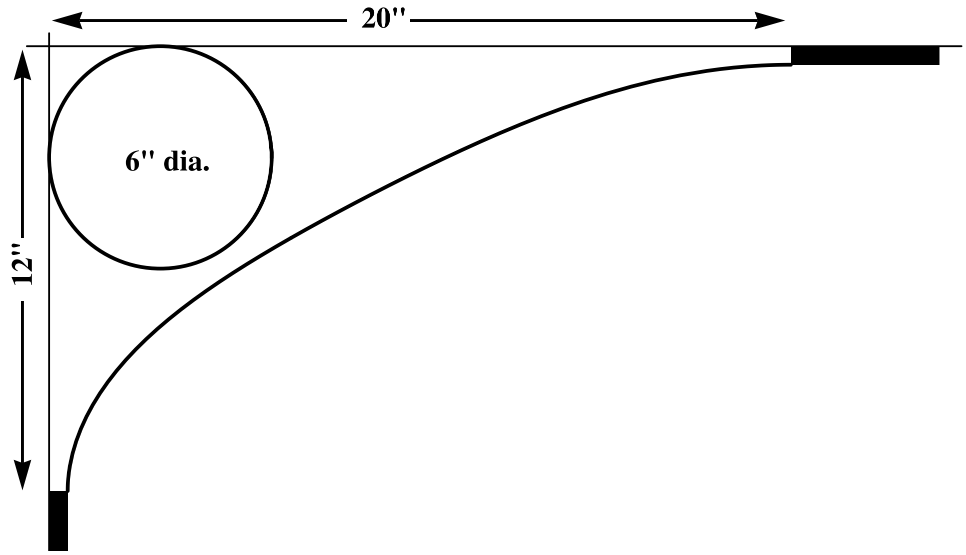


Figure 4.22. Coving Design

A coving designed to smoothly blend the joint between the wall and the ceiling, and hide a heating duct.

Figure Name & Figure Number	Functional	Time in Seconds	Number of Iterations	Final MEC Functional	Final MVC Functional
Wicket	MVC	1.2	15	3.141	2.176×10^{-6}
Fig. 4.2	MEC	0.6	6	2.871	1.596
G^2 Box	MVC	2.6	6	7.247	26.15
Fig. 4.10①	MEC	6.4	13	6.291	718.6
G^1 Box	MVC	4.7	7	6.283	7.09×10^{-6}
Fig. 4.10②	MEC	15.5	24	6.268	1.027
Woodford	MVC	75.0	137	2.551	1.951
Fig. 4.11	MEC	18.4	31	2.248	2.714
Space Curve	MVC	12.3	90	5.098	4.597
Fig. 4.16	MEC	4.4	29	5.074	5.270
Peters' Helix	MVC	29.3	27	4.913	0.089
Fig. 4.21	MEC	17.3	15	4.658	0.156

Table 4.1. Timings for Curve Examples—MVC&MEC

Each optimization was run until the infinity norm of the gradient was less than 0.0005.

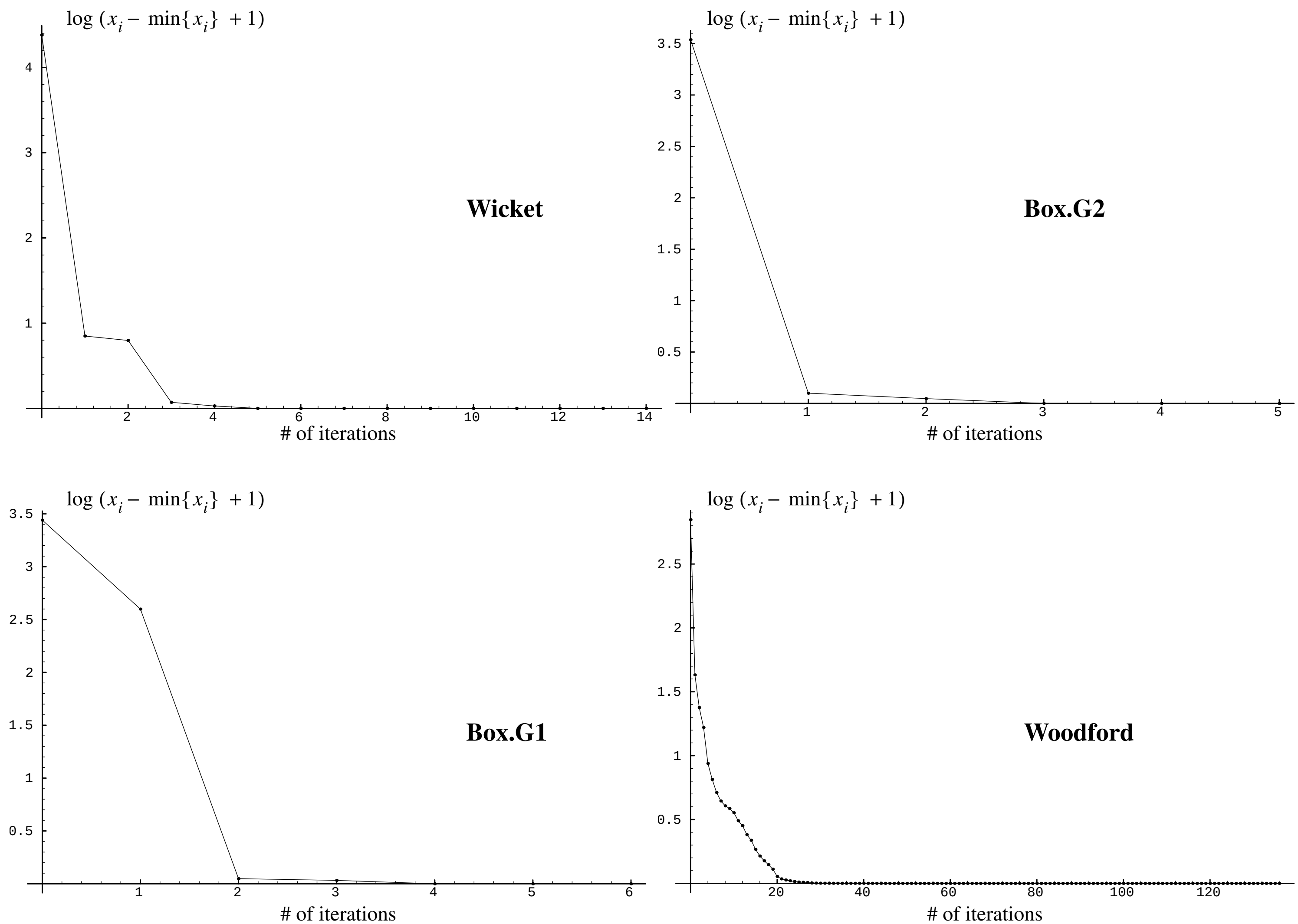


Figure 4.23. Log Plots of MVC Convergence

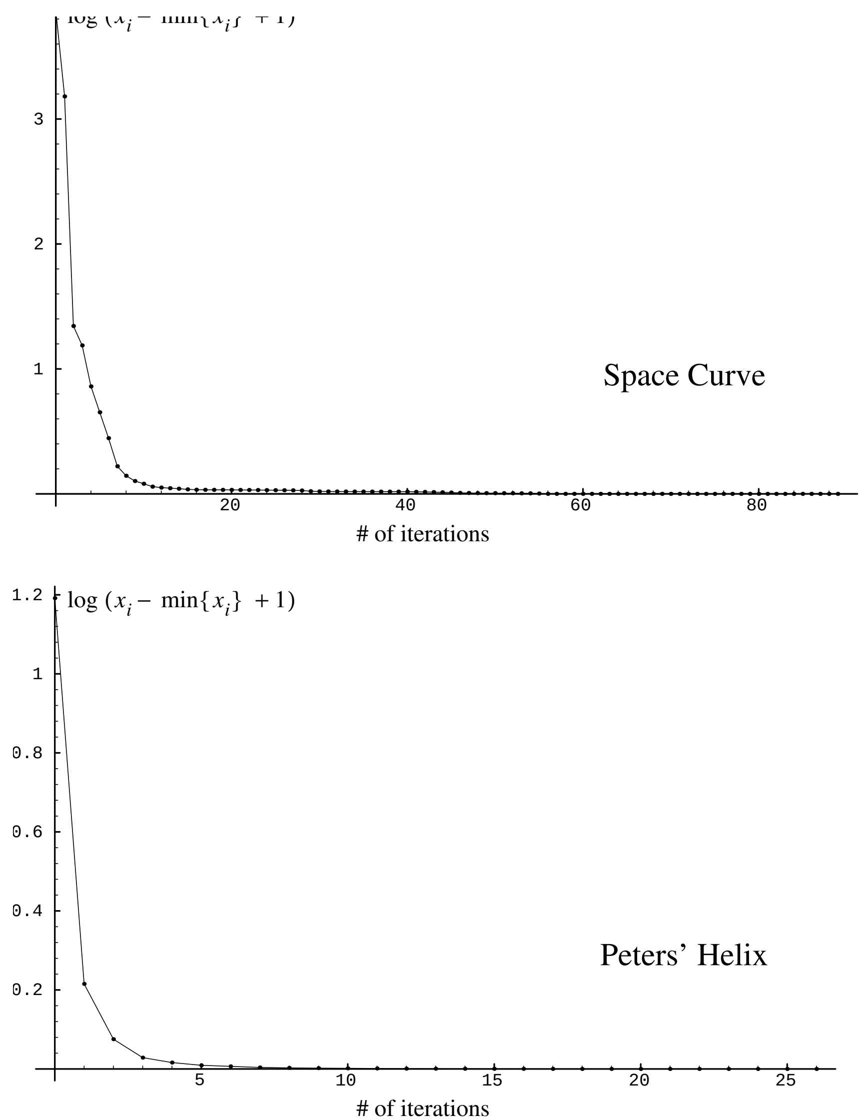


Figure 4.24. Log Plots of MVC Convergence (cont.)