Supplementary Material: Part Type Kinematics and Initialization for "Motion-Guided Mechanical Toy Modeling"

1 Introduction

This supplement describes the parameters, kinematic behavior, and initialization for all mechanical parts used in our paper. Four aspects of each mechanical parts are included:

- **Shape parameters.** The motion of the part is controlled by its shape parameters.
- **Constraints on shape parameters.** Certain constraints should be maintained to ensure the mechanical part functions properly.
- **Kinematic simulation.** The position and orientation, called the *local frame*, of each rigid component in the mechanical part must be determined at each simulation time t, given its shape parameters.
- **Shape parameters initialization.** Shape parameters must be initialized at the start of optimization to approximate the target handle trajectory specified by the designer.

Since the driving motion is a periodic rotation of the driving axis, we require that the input motion of a feature object also be periodic. Denote the number of frames in one period as n. Initialization of each part is based on one period of the target trajectory $\{\mathbf{p}_i, i = 1..n\}$ sampled from the initial handle on the feature object. The free end point of each part, denoted $\mathbf{e}_i$, is kinematically defined for each part type, and connects to this handle to realize the target motion.

Note: This supplement uses the notation $\mathbf{p}$ for the user-specified target trajectory of the initial handle (denoted $\hat{\mathbf{p}}^*$ in the main paper) and $\mathbf{e}$ for the kinematically-defined trajectory of the part's end effector (denoted $\mathbf{p}$ in the main paper). Since the supplement only considers individual parts rather than assemblies, there is no need to doubly index these points. Index i here refers to the animation frame. The supplement also uses n for the number of animation frames (denoted N in the main paper).

Common parameters: The following parameters are common across all part types:

- $\mathbf{o}$: mechanical part's rotational center. The part's local coordinate system uses this point as the origin.
- $\mathbf{a}$: mechanical part's rotation axis direction. In our system, it is always equal to the canonical $\mathbf{X}$ axis.
- h: distance of part's driving axis (including $\mathbf{o}$) below the upper wall of the assembly, along the canonical $\mathbf{Y}$ axis.
- n: period of the mechanism in number of frames.
- m: period of the driving axis in number of frames.
- ω: angular velocity of the driving axis.
- F: frame rate.
- t_i: simulation time of the i-th frame, $t_i = (i-1)/F, i = 1..n$.
- r^*: mean radius; the default is $\frac{1}{10}$ of the diagonal length of the assembly box.

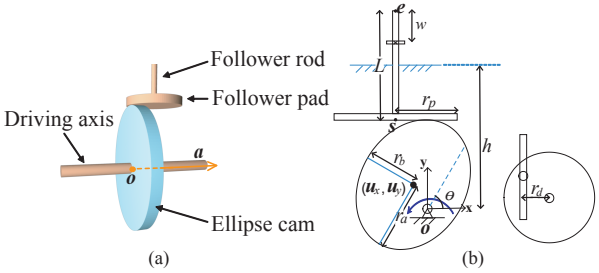


Figure 1: *Ellipse cam. (a) 3D view. (b) Schematic view.*

2 Ellipse Cam

Description

The ellipse cam part is a cam/follower system whose cam contour is an ellipse. Refer to Figure 1a. The follower pad translates up and down in $\mathbf{Y}$, and also possibly rotates around the $\mathbf{Y}$ axis, as the driving cam rotates around the $\mathbf{X}$ axis.

Parameters

Parameters are defined in the part's local coordinate system, as shown in Figure 1b:

- L: length of the follower rod.
- r_p: radius of the follower pad.
- w: minimum length of the follower rod above the upper wall of the assembly box.
- r_d: distance from the follower pad center to the projected cam center.
- r_a: cam's long axis.
- r_b: cam's short axis.
- θ: cam's phase.
- $\mathbf{u}_x, \mathbf{u}_y$: offset of ellipse center from rotation center.

Constraints

1. The rotation center should be inside the ellipse.
2. The follower pad must be wide enough; i.e., $r_p > r_d$.
3. $w < L$.

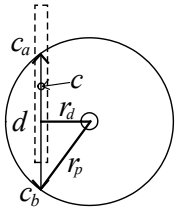
Kinematic simulation

The local frame of the ellipse cam and the follower pad must be computed at each simulation time t_i.

Local frame of cam: The cam rotates around the $\mathbf{X}$ axis, centered at the point $\mathbf{o}$, which forms the origin of the local coordinate system. The rotational angle is given by $\theta_i = \theta + \omega t_i$.

Local frame of follower: The local frame of the follower pad is determined after the local frame of the cam, since the cam drives the follower pad's height and rotation. The computation is detailed in the following two sections. Pad rotation only occurs when the distance from the center of the pad to the ellipse cam, r_d, is non-zero.

Pad height: As shown in the right inset, the contact point $\mathbf{c}$ lies on the chord c_a, c_b of length $d = 2\sqrt{r_p^2 - r_d^2}$. It is either the highest point on the ellipse or the projection of one of the chord end points c_a, c_b onto the ellipse. Points on a canonical ellipse can be represented in terms of a parametric angle ϕ, as $(r_a \cos(\phi), r_b \sin(\phi))^T$. These are then rotated around an arbitrary pivot point $(\mathbf{u}_x, \mathbf{u}_y)$ by an angle θ, representing rotation of the cam's driving axis. The ellipse's kinematic configuration is therefore given by:



$$P(\phi; \mathbf{u}_x, \mathbf{u}_y, \theta) = R(\theta)\, T(\mathbf{u}_x, \mathbf{u}_y) \begin{pmatrix} r_a \cos\phi \\ r_b \sin\phi \end{pmatrix} \quad (1)$$
$$= \begin{pmatrix} (r_a \cos\phi + \mathbf{u}_x)\cos\theta - (r_b \sin\phi + \mathbf{u}_y)\sin\theta \\ (r_a \cos\phi + \mathbf{u}_x)\sin\theta + (r_b \sin\phi + \mathbf{u}_y)\cos\theta \end{pmatrix}$$

where $T(x,y)$ represents the 2D translation from the origin to the point (x,y), and $R(\theta)$ represents 2D counterclockwise rotation around the origin by an angle of θ. Maximizing the y coordinate of P over all ϕ as a function of $(\theta, \mathbf{u}_x, \mathbf{u}_y)$ yields the equation:

$$\frac{\partial P_y}{\partial \phi} = \sin\phi(-r_a \sin\theta) + \cos\phi(r_b \cos\theta) = 0 \quad (2)$$

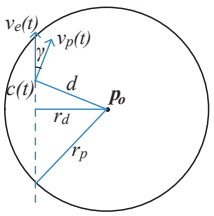
Its solution, ϕ^* where $\tan\phi^* = \frac{r_b \cos\theta}{r_a \sin\theta}$, can then be substituted into (1) to obtain the coordinates of $\mathbf{c}$.

If this point is not on the segment c_a, c_b, then the contact point $\mathbf{c}$ is the projection of either c_a and c_b onto the ellipse, whichever is higher. We can compute this projection by computing ϕ from the endpoint's x coordinate and then substituting it into Equation 1 to find P_y.

The height of the free end point is $\mathbf{c}_y + L$. If $\mathbf{c}_y + (L - w)$ is less than the height of the assembly box, then the fixed joint installed at w will stop at its upper wall. The end point therefore can not attain a height below $h + w$, where h is the distance of the driving axis below the upper wall of the assembly box. This fixed joint enables pauses in the handle motion, where it maintains this lowest height over part of the animation cycle. We obtain the final position of the cam's end effector via

$$\mathbf{e}_y = \max(\mathbf{c}_y + L, h + w). \quad (3)$$

Pad rotation: The follower pad rotates around the follower rod in terms of a 1D angular parameter denoted $\alpha(t)$. We denote its derivative (angular speed) as $\dot{\alpha}(t)$. Denote the moving contact point of the ellipse cam as $\mathbf{c}(t)$ and the height of the pad as $g(t)$. Then the linear speed $v_p(t)$ of the contact point $\mathbf{c}(t)$ on the pad is given by $v_p(t) = \dot{\alpha}(t)\, d$, where d is distance between the pad center, $\mathbf{p}_o$, and the contact point, $\mathbf{c}$. Its direction is perpendicular to $\mathbf{p}_o - \mathbf{c}$, as indicated in the right inset. Similarly, the linear speed of the contact point on the elliptical cam is $v_e(t) = \omega\, g(t)$. The magnitude of v_p should equal the magnitude of the component of $\mathbf{v}_e$ along $\mathbf{v}_p$, or

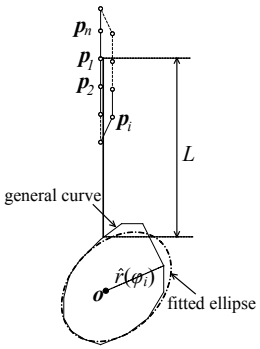


$$v_p(t) = v_e(t)\cos(\gamma) \quad (4)$$

where γ is the angle between the two velocity vectors. This cosine can be computed via $\cos(\gamma) = r_d/d$. Given $v_p(t)$, then $\dot{\alpha}(t) = v_p(t)/d$ and the current rotation angle α is obtained through numerical integration.

Initialization

The ellipse cam can be used to realize 2D linear, 2D oscillating, or 3D helical motion. Target motion is represented by the trajectory, $\{\mathbf{p}_i, i = 1..n\}$, of the handle controlled by this part, sampled at frame times $\{t_i, i = 1..n\}$.



Initialization for linear motion: We first compute the cam contour as a general, sampled, closed curve based on one period of input motion. The curve is represented by $\{\mathbf{o}, \hat{r}(\phi)\}$, where $\mathbf{o}$ is the cam's rotation center as described above, ϕ is a parametric angle, and $\hat{r}(\phi)$ is the distance of the point on the general curve to the center. We perform the following steps:

1. Set $r_d = 0$, since no rotation is present (we want linear motion only). Set r_p to a default ratio, 0.003, of the assembly box diagonal length. Set $w = 0$.
2. Compute the initial follower length, L. Compute the distance d_1 of $\mathbf{p}_1$, the handle position at the beginning of the motion, to the driving axis. Set $L = 0.8\, d_1 - r^*$.
3. Compute $\mathbf{o}$. We fit a straight line to the handle trajectory $\{\mathbf{p}_i, i = 1..n\}$ and then compute the closest point on the driving axis to this line.
4. Compute $\hat{r}(\phi)$. At each frame i, $\phi_i = \omega t_i$ and $\hat{r}(\phi_i) = \mathbf{p}_i.y - \mathbf{o}_y - L$.
5. Compute $r_a, r_b, \mathbf{u}_x, \mathbf{u}_y, \theta$ based on the contour $\{\hat{r}(\phi_i)\}$. Calculate its OBB (oriented bounding box) and set r_a, r_b to be half its height and width. Set θ to the rotation angle from the canonical local frame to the local frame of the computed OBB. Set $(\mathbf{u}_x, \mathbf{u}_y)$ to the difference between the OBB's center and $\mathbf{o}$.

Initialization for oscillating motion: We fit a line to the oscillating handle trajectory, and then invoke the previous initialization procedure. We then connect the handle to the driven feature component in one of two ways. If the component is on a kinematic chain with more than one joint, we make a direct attachment. If the kinematic chain contains only one joint, a slot is required in the feature component to let the end point slide freely as its local position changes.

Figure 2 shows how we can determine the necessary slot. We use the following steps:

1. Compute $\mathbf{e}_1$, the initial position of the free end point, as a point on the sliding slot.
2. Compute $\mathbf{c}^*$, the intersection of the cam follower and the driven component $\mathbf{C}$.

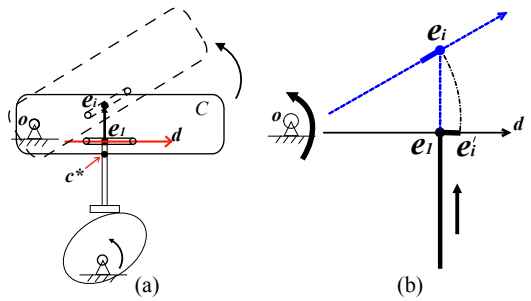


Figure 2: *Sliding slot. (a) 2D view. (b) Schematic view.* $\mathbf{e}_i$ *is the position of the part end at frame i.* $\mathbf{e}'_i$ *is its transformation back into the local frame of the feature component at frame 1.*

3. Compute the driven component's tangent plane at $\mathbf{c}^*$.
4. Compute the intersection line of this tangent plane with the XY plane of the oscillating motion.

The slot direction is equal to this line direction, $\mathbf{d}$.

When driving a kinematic chain with a sliding slot, $\mathbf{e}_i$ must lie on the line $l(\mathbf{e}_1, \mathbf{d})$, in the local coordinate system of the feature component. This can be achieved with an IK target that constrains the distance between the point $\mathbf{e}'_i$ and that line to be 0, where $\mathbf{e}'_i$ is the transformation of $\mathbf{e}_i$ back to the local coordinate system at frame 1. The sliding slot's length can be determined by finding the two extreme points of $\mathbf{e}'_i$ along $\mathbf{d}$.

A slotted connection also requires that the (possibly updated) handle position must lie entirely within the feature component it connects to, across the entire animation.

Initialization for helical motion: Helical motion can be decomposed into rotational motion of the follower pad and linear motion of the follower rod. We perform the following steps to initialize the ellipse cam for helical motion:

1. Compute the central axis for the helical motion. This is represented by the line $l(\mathbf{q}, \mathbf{d})$, where $\mathbf{q}$ is a point on the line and $\mathbf{d}$ is its direction vector. We compute $\mathbf{q}$ and $\mathbf{d}$ by minimizing

$$E = V\left(\{d\left(\mathbf{p}_i, l(\mathbf{q}, \mathbf{d})\right)\}\right)$$

subject to $||\mathbf{d}|| = 1$. Here, V represents variance:

$$V(\{x_i\}) = \sum_{i=1}^{n} \left(x_i - \frac{1}{n} \sum_{j=1}^{n} x_j \right)^2,$$

and d represents distance from a point to a line:

$$d\left(\mathbf{p}_i, l(\mathbf{q}, \mathbf{d})\right) = \left\| \mathbf{p}_i - \mathbf{q} - \left((\mathbf{p}_i - \mathbf{q}) \cdot \mathbf{d} \right) \mathbf{d} \right\|.$$

To start the optimization, $\mathbf{q}$ is initialized to the mean of $\{\mathbf{p}_i\}$, and $\mathbf{d}$ is initialized to the $\mathbf{Y}$ axis.

2. Analyze the rotational component. First compute the projection of the $\mathbf{p}_i$ onto the helix's central axis, yielding $\bar{\mathbf{p}}_i$. Then extract the directions

$$\mathbf{v}_i = \frac{\mathbf{p}_i - \bar{\mathbf{p}}_i}{\|\mathbf{p}_i - \bar{\mathbf{p}}_i\|}$$

to compute the rotational phase at frame i. Setting the phase at $\mathbf{p}_1$ to 0, the phase of $\mathbf{p}_i$ is the angle γ_i between $\mathbf{v}_i$ and $\mathbf{v}_1$.

3. Initialize the shape parameters based on the detected linear motion. Use $r_d = 0$ for this initialization.
4. Analyze the period m' of the rotational component by detecting the frame where γ_i returns to 0. Set $r_d = r_a m'/m$, where m is the period of the driving axis and r_a is the length of the ellipse's long axis. Set $r_p = r_d + T$, where T is the cam thickness.

Local optimization of the parameters: The shape parameters can be further optimized by minimizing the error between the actual and target trajectories of the part's end point,

$$E = \sum_{i=1}^{n} \|\mathbf{e}_i - \mathbf{p}_i\|^2, \tag{5}$$

subject to the ellipse cam's constraints.

Gradient descent is used to minimize E. The part constraints can be violated in its 1D search step; we simply project the parameters back to their closest feasible point. Though this simple method could be improved, it works adequately for initialization.

Kinematic simulation of the ellipse cam exhibits discontinuous state switches; e.g., the contact between the cam and the follower pad is either the highest point on the cam or the projection of an end point of the contact chord onto the ellipse. This makes it difficult to formulate a closed-form function for the actual trajectory of $\mathbf{e}$. We thus use a numerical gradient to optimize the initialized shape parameters.

3 Double Cam

Description

The double ellipse cam generates more complex motion than the single ellipse cam from the previous section. It consists of two elliptical cams and a single follower with a pad, as shown in Figure 3. The two cams are placed at different sides of the follower pad. When driven, the two cams alternately touch the follower pad to reverse its rotational direction in one cycle of driving axis rotation.

Parameters

- L: length of the follower rod.
- r_p: radius of the follower pad.
- w: minimum length of the follower rod above the upper wall of the assembly box.
- r_d: distance from the follower pad center to cam A's projected center.
- r_a: cam A's long axis.
- r_b: cam A's short axis.
- θ: phase of cam A.
- $\mathbf{u}_x, \mathbf{u}_y$: offset of cam A's ellipse center from its rotation center.
- r'_d: distance from the follower pad center to cam B's projected center.
- r'_a: cam B's long axis.
- r'_b: cam B's short axis.
- θ': phase of cam B.

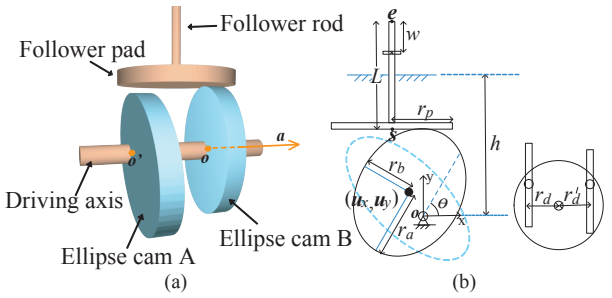


Figure 3: *Double ellipse cam. (a) 3D view. (b) Schematic view.*

- $\mathbf{u}'_x, \mathbf{u}'_y$: offset of cam B's ellipse center from its rotation center.

Constraints

1. Both cam rotation centers should be inside their respective ellipses.
2. The follower pad must be wide enough; i.e., $r_p > \max(r_d, r'_d)$.
3. $w < L$.
4. The two cams do not collide; $\min(r_d, r'_d) > T$, where T is the cam thickness.

Kinematic simulation

Local frame of cams: The cams' positions are fixed on the driving axis. Their rotation angles θ_i, θ'_i at the i-th frame can be computed via the phase θ, θ' and angular velocity ω of the driving axis: $\theta_i = \theta + \omega t_i$, $\theta'_i = \theta' + \omega t_i$.

Local frame of follower: The follower's local frame depends on which cam is touching its pad at time t_i. We compute each ellipse cam's highest point in the same way as with the single cam, yielding the functions $g(t)$ and $g'(t)$. When $g > g'$, cam A drives the follower; otherwise, cam B does. If the follower is blocked by the upper wall of the assembly box via the fixed joint installed at w, neither cam touches the pad. In this case, its local frame remains constant until a cam resumes contact.

Initialization

Initialization for helical motion: The procedure is similar to that for the single ellipse cam, except that it segments the rotational motion into two components. We compute frame indices, i^*, where the rotational phase, $\{\gamma_i\}$, switches the sign of its derivative. Note that a jump of 2π must be handled in this segmentation because the source motion is periodic. We then assemble these monotonic pieces into two components: one where γ increases and the other where it decreases. The handle trajectory corresponding to the increasing component initializes the first cam; the second cam is initialized from the decreasing component. Note that these components form an incomplete target for each cam's contour, because only one cam drives at a time.

Given the incomplete handle trajectory for each cam, we compute its respective cam contour target as in step 4 for linear motion initialization for the single cam. We fit an ellipse to the resulting $\hat{r}(\phi)$, since the OBB from an incomplete curve might not pass through the rotational center on the driving axis. We then compute an overall period for each cam's rotation. We choose two points in a segment and calculate the angular speed ω by dividing angular difference by time difference; the period is $2\pi/\omega$. We then compute r_d and r'_d based on the corresponding overall angular period required. We set $r_p = \max(r_d, r'_d) + T$, where T is the cam thickness. Finally, we choose the position, $\mathbf{o}$ or $\mathbf{o}'$, along the driving axis in order to force its distance to the follower pad center to be r_d and r'_d, respectively.

Initialization for linear motion: When the part's free end point is connected to the driven component by a translational joint, the double cam produces linear motion. Rotation of the follower rod can be ignored, because of the ball joint implied by this connection. We perform the following steps to initialize for the linear motion case:

1. Initialize cam A according to the input linear motion.
2. Set cam B's shape parameters to be identical to cam A, except in phase where $\theta' = \theta + \pi/2$.
3. Set $r_d = r'_d = r_b$, and $r_p = r_d + T$, where T is the cam thickness.

This is a rough initialization, which is improved via shape parameter optimization as described in the main paper.

Local optimization of the parameters: The method is essentially the same as for the ellipse cam. Cam-switching makes the kinematic motion discontinuous; we handle this by taking numerical gradients.

4 Snail Cam

Description

This cam's contour has a snail-like shape as illustrated in Figure 4a. It turns rotary motion into linear. The difference between it and other cam types is that the motion of its follower consists of a slow lift and a sudden drop. To produce this motion, the polar representation of its cam contour, $r(\phi)$, should be a piecewise linear function, as shown in Figure 5. The increasing and subsequent decreasing segments form an extrusion, called a *lobe*, on the contour. The slope of this lobe's increasing segment should be limited to ensure the cam works smoothly.

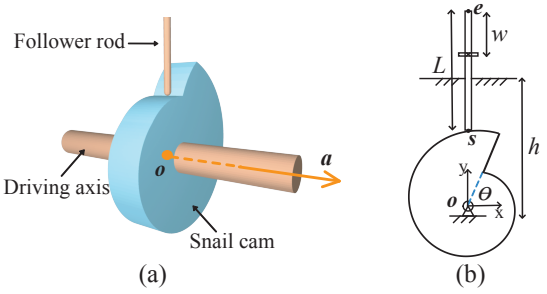


Figure 4: *Snail cam. (a) 3D view. (b) Schematic view.*

Our system also supports a multi-lobed snail cam type. Figure 5 shows a single-lobed and a 3-lobed snail cam.

Parameters

- L: length of the follower.
- w: minimum length of the follower above the upper wall of the assembly box.
- θ: initial phase of the snail cam.

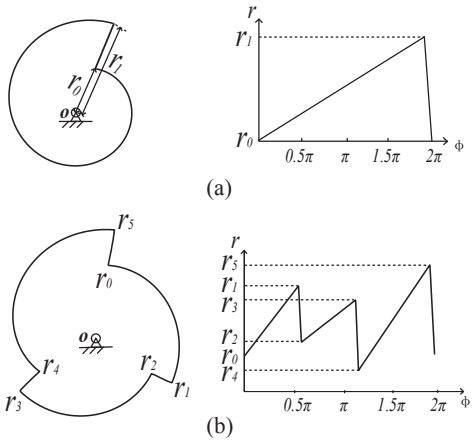


Figure 5: *Snail cam. (a) Left: Contour of a single-lobed snail cam. Right: Its polar representation. (b) Left: Contour of a 3-lobed snail cam. Right: Its polar representation.*

- $r(\phi)$: piecewise linear function representing the cam's contour in polar coordinates.

Constraints

1. $r(\phi) > 0, \forall \phi \in [0, 2\pi]$.
2. $0 < \frac{r(\phi_i) - r(\phi_{i-1}))}{\phi_i - \phi_{i-1}} < \frac{v}{\omega}$ where $\{r(\phi_{i-1}), r(\phi_i)\}$ denotes the increasing segment for the i-th lobe. v is a threshold on the rising velocity of the follower. It is set to 10 cm/s in our implementation.
3. $w < L$.

Kinematic simulation

Local frame of the cam: Rotation angle is $\theta_i = \theta + \omega t_i$.

Local frame of the follower: The follower's motion is purely linear, with no rotation. Its height is controlled by the snail cam. At simulation time t_i, if the snail cam contacts the follower, its height is $r(\theta + \omega t_i)$. Otherwise, the follower is blocked by the fixed joint installed at w. So its height is $\max(r(\theta + \omega t_i), h - (L - w))$. where h is the distance of the assembly box's upper wall to the cam's rotation center.

Initialization

Initialization for linear motion: The procedure is similar to the case of an ellipse cam. Given the target trajectory $\{\mathbf{p}_i, i = 1...n\}$ of the follower's end point, we perform the following steps:

1. Set w and θ to 0.
2. Compute the initial follower length, L, as in ellipse cam initialization.
3. Compute the rotation center $\mathbf{o}$ as in ellipse cam initialization.
4. Compute $\hat{r}(\phi_i)$. At each frame i, $\phi_i = \omega t_i$ and $\hat{r}(\phi_i) = \mathbf{p}_i.y - \mathbf{o}_y - L$.
5. Smooth the curve $\hat{r}(\phi)$ using Laplacian smoothing [Desbrun et al. 1999].
6. Detect all local extrema, $\{(\phi_j, r_j), j = 1..k\}$, in the smoothed curve $\hat{r}(\phi_i)$. The final snail cam contour $r(\phi)$ is then reconstructed from these detected extrema via:

$$r(\phi) = \frac{\phi_{j+1} - \phi}{\phi_{j+1} - \phi_j} r_j + \frac{\phi - \phi_j}{\phi_{j+1} - \phi_j} r_{j+1}, \phi \in [\phi_j, \phi_{j+1}]. \quad (6)$$

Initialization for oscillating motion: See the corresponding description for the ellipse cam.

Local optimization of the parameters: See the corresponding description for the ellipse cam. Kinematic behavior of the snail cam's end point is given by

$$\mathbf{e}_i = \mathbf{o} + \mathbf{Y} \max(h + w, r(\theta + \omega t_i) + L)$$

where the function $r(\phi)$ is defined in Equation 6.

5 Crank-Slider

Description

The crank-slider part drives a slider rod via a rotating crank as shown in Figure 6. The slider is constrained to pass through a hole at a fixed point on the upper wall of the assembly box, indicated by $\mathbf{q}$ in Figure 6b. The crank's driving axis direction is $\mathbf{X}$ and its center of rotation is at $\mathbf{o}$, located a fixed distance h below the box's upper wall along the $\mathbf{Y}$ direction.

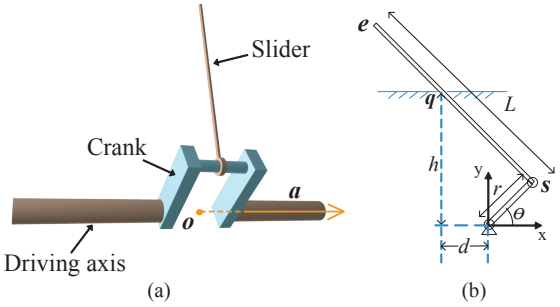


Figure 6: *Crank-slider. (a) 3D view. (b) Schematic view.*

Parameters

- r: crank radius.
- L: slider length.
- θ: initial phase of the crank-slider.
- d: projection of $\mathbf{q} - \mathbf{o}$ onto the $\mathbf{X}$ axis, $d = (\mathbf{q} - \mathbf{o}) \cdot \mathbf{X}$.

Constraints

1. The slider's end point should extend above the upper wall even at its lowest position: $L - r > \|\mathbf{o} - \mathbf{q}\|$.

Kinematic simulation

Local frame of the crank: The crank's motion is a simple 2D rotation around $\mathbf{X}$ centered at $\mathbf{o}$, in terms of angle $\theta_i = \theta + \omega t_i$. The rotation center $\mathbf{o}$ forms the origin of the part's local coordinate system.

Local frame of the slider: In the xy frame with center at $\mathbf{o}$, as shown in Figure 6b, the starting point $\mathbf{s}_i$ where the slider attaches to

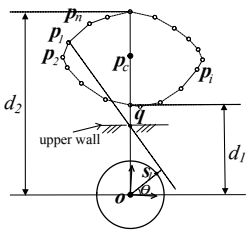
the crank is given by $\mathbf{s}_i = (r\cos(\theta_i), r\sin(\theta_i))$. The coordinates of $\mathbf{q}$ in this coordinate frame are (d,h). The slider's end point in the local coordinate system is given by

$$\mathbf{e}_i = \mathbf{s}_i + L\frac{\mathbf{q}-\mathbf{s}_i}{\|\mathbf{q}-\mathbf{s}_i\|}. \tag{7}$$

Given the position of these two end points on the slider rod, $\mathbf{s}_i$ and $\mathbf{e}_i$, its local frame is easily determined by noting that it does not rotate about its own axis.

Initialization

The trajectory of $\mathbf{e}$ over one animation cycle forms a closed planar curve. Given the target trajectory $\{\mathbf{p}_i, i = 1..n\}$, we initialize the crank-slider's parameters using the following steps:



1. Compute $\mathbf{o}$. Compute the center of the target trajectory $\mathbf{p}_c$ and find its projection along the driving axis.

2. Compute L and r. Find the closest and furthest distance of the input trajectory $\{\mathbf{p}_i\}$ to $\mathbf{o}$, denoted by d_1 and d_2. Since the closest and furthest distance of $\mathbf{e}$ to $\mathbf{o}$ is $L-r$ and $L+r$ respectively, according to Equation 7, set $r = (d_2 - d_1)/2$ and $L = (d_1 + d_2)/2$.

3. Initialize d. Compute $\mathbf{q}$ as the intersection of the line from $\mathbf{o}$ to $\mathbf{p}_c$ with the assembly box's upper wall. Set $d = (\mathbf{q}-\mathbf{o})\cdot\mathbf{X}$.

4. Compute the initial phase θ. We determine $\mathbf{s}_1$ by finding the point on the circle $(\mathbf{o}, r)$ nearest the line passing through $\mathbf{q}$ and $\mathbf{p}_1$. θ is the angle between the vector $\mathbf{s}_1 - \mathbf{o}$ and the $\mathbf{X}$ axis.

Local optimization of the parameters: As with the other part types, the initialized parameters are further optimized by minimizing Equation 5, in terms of its own end point kinematics and subject to its constraints. We list the derivatives required to evaluate the required objective gradient:

$$\frac{\partial E}{\partial x} = 2\sum_{i=1}^{n}(\mathbf{e}_i - \mathbf{p}_i)^T\frac{\partial \mathbf{e}_i}{\partial x}. \tag{8}$$

where x can be any of the part's shape parameters, r, L, θ, or d. By Equation 7,

$$\frac{\partial \mathbf{e}_i}{\partial x} = \begin{cases} \dfrac{\partial \mathbf{s}_i}{\partial x} + L\dfrac{\partial}{\partial x}\left(\dfrac{\mathbf{q}-\mathbf{s}_i}{\|\mathbf{q}-\mathbf{s}_i\|}\right), & x = r, \theta, \text{ or } d \\ \dfrac{\mathbf{q}-\mathbf{s}_i}{\|\mathbf{q}-\mathbf{s}_i\|}, & x = L. \end{cases}$$

By the chain rule,

$$\frac{\partial}{\partial x}\left(\frac{\mathbf{q}-\mathbf{s}_i}{\|\mathbf{q}-\mathbf{s}_i\|}\right) = \frac{1}{\|\mathbf{q}-\mathbf{s}_i\|}M\left(\frac{\partial \mathbf{q}}{\partial x} - \frac{\partial \mathbf{s}_i}{\partial x}\right),$$

where

$$M = I - \frac{\mathbf{q}-\mathbf{s}_i}{\|\mathbf{q}-\mathbf{s}_i\|}\left(\frac{\mathbf{q}-\mathbf{s}_i}{\|\mathbf{q}-\mathbf{s}_i\|}\right)^T.$$

The derivatives required to compute the above are: $\partial \mathbf{s}_i/\partial r = (\cos\theta_i, \sin\theta_i)$, $\partial \mathbf{s}_i/\partial \theta = (-r\sin\theta_i, r\cos\theta_i)$, $\partial \mathbf{s}_i/\partial d = (0,0)$, $\partial \mathbf{q}/\partial r = (0,0)$, $\partial \mathbf{q}/\partial \theta = (0,0)$, and $\partial \mathbf{q}/\partial d = (1,0)$.

6 Quick-Return

Description

The quick-return is illustrated in Figure 7. Like the crank-slider, the quick-return is also a slider driven by a crank, It also converts rotary motion at the crank into oscillating motion at the end point of the slider. But it differs from the crank-slider in that the slider rod's starting point is not fixed on the moving crank. Instead, it is attached to a hinge joint installed inside the assembly box at a fixed location $\mathbf{s}$. The crank attaches to the slider with a translational joint at an intermediate point $\mathbf{f}$. This is simply a slot along which the rod slides and where the crank drives its angle at the hinge.

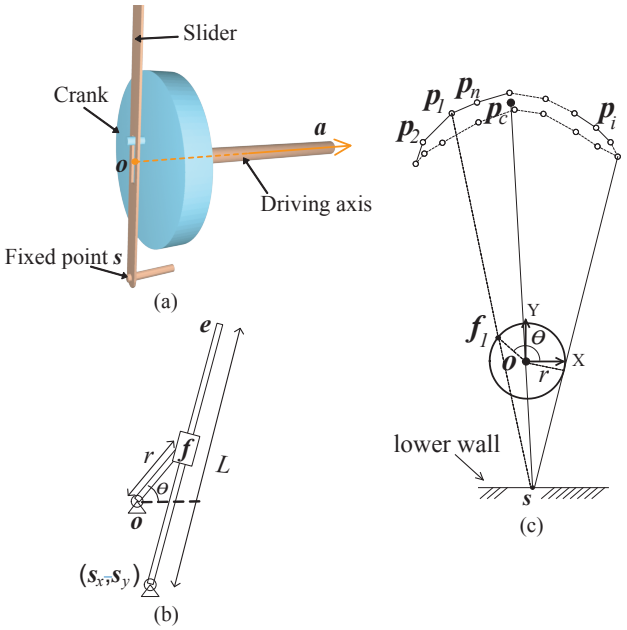


Figure 7: *Quick-return. (a) 3D view. (b) Schematic view. (c) Initialization schematic.*

Parameters

- r: crank radius.
- L: slider length.
- θ: initial phase of the quick-return.
- $(\mathbf{s}_x, \mathbf{s}_y)$: fixed hinge point in the quick-return's local frame.

Constraints

1. the fixed point of the quick return should be below the crank: $\|\mathbf{s}-\mathbf{o}\| > r$
2. the translational joint between the crank and the slider never exceed the slider: $L > 2r$
3. the fixed point $\mathbf{s}$ is above the lower wall of the assembly box

Kinematic simulation

Local frame of the crank: The crank's motion is a 2D rotation around $\mathbf{X}$, centered at $\mathbf{o}$. The rotation angle is $\theta_i = \theta + \omega t_i$.

Local frame of the slider: As with the crank-slider, the local frame of the slider at the i-th frame can be computed given its (fixed) starting point, $\mathbf{s}$, and (moving) end point, $\mathbf{e}_i$. The slider's end point in the local coordinate system is given by:

$$\mathbf{e}_i = \mathbf{s} + L\frac{\mathbf{f}_i - \mathbf{s}}{||\mathbf{f}_i - \mathbf{s}||},$$

where $\mathbf{f}_i = (r\cos(\theta_i), r\sin(\theta_i))$.

Initialization

The quick-return is used as a candidate part to realize a desired motion when that motion is approximately oscillatory. Refer to Figure 7c. Given the input handle trajectory, $\{\mathbf{p}_i, i = 1..n\}$, we initialize a quick-return for that handle by performing the following steps:

1. Compute the quick-return's rotation center $\mathbf{o}$ as the projection of the trajectory center, $\mathbf{p}_c$, along the driving axis.
2. Compute $\mathbf{s}$, as the intersection between the line through $\mathbf{p_c}$ and $\mathbf{o}$, and the lower wall of the assembly box.
3. Initialize r to be the largest distance from $\mathbf{o}$ to the set of lines passing through $\mathbf{s}$ and the $\{\mathbf{p}_i\}$.
4. Initialize the slider length L to be the average of $\{\|\mathbf{s} - \mathbf{p}_i\|, i = 1..n\}$.
5. Compute the initial phase of the crank θ. Find $\mathbf{f}_1$ as the point on the circle $(\mathbf{o}, r)$ nearest to the line through $\mathbf{s}$ and $\mathbf{p}_1$. Set θ to the angle between $\mathbf{f}_1 - \mathbf{o}$ and the canonical $\mathbf{X}$ axis.

Local optimization of the parameters: Local optimization is similar to the case of the crank-slider; see Equations 5 and 8. In the case of the quick-return, the end point position $\mathbf{e}_i$ is given above, and the partial derivatives x can be with respect to the parameters r, L, θ, $\mathbf{s}_x$, or $\mathbf{s}_y$. The quick-return's kinematics equation yield:

$$\frac{\partial \mathbf{e}_i}{\partial x} = \begin{cases} \frac{\partial \mathbf{s}}{\partial x} + L\frac{\partial}{\partial x}\left(\frac{\mathbf{f}_i - \mathbf{s}}{\|\mathbf{f}_i - \mathbf{s}\|}\right), & x = r, \theta, \mathbf{s}_x \text{ or } \mathbf{s}_y \\ \frac{\mathbf{f}_i - \mathbf{s}}{\|\mathbf{f}_i - \mathbf{s}\|}, & x = L \end{cases} \quad (9)$$

By the chain rule,

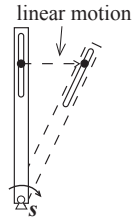
$$\frac{\partial}{\partial x}\left(\frac{\mathbf{f}_i - \mathbf{s}}{\|\mathbf{f}_i - \mathbf{s}\|}\right) = \frac{1}{\|\mathbf{f}_i - \mathbf{s}\|} M \left(\frac{\partial \mathbf{f}_i}{\partial x} - \frac{\partial \mathbf{s}}{\partial x}\right),$$

where

$$M = I - \frac{\mathbf{f}_i - \mathbf{s}}{\|\mathbf{f}_i - \mathbf{s}\|}\left(\frac{\mathbf{f}_i - \mathbf{s}}{\|\mathbf{f}_i - \mathbf{s}\|}\right)^T.$$

The derivatives required to compute the above are: $\partial \mathbf{f}_i/\partial r = (\cos\theta_i, \sin\theta_i)$, $\partial \mathbf{f}_i/\partial \theta = (-r\sin\theta_i, r\cos\theta_i)$, $\partial \mathbf{f}_i/\partial \mathbf{s}_x = (0,0)$, $\partial \mathbf{f}_i/\partial \mathbf{s}_y = (0,0)$, $\partial \mathbf{s}/\partial r = (0,0)$, $\partial \mathbf{s}/\partial \theta = (0,0)$, $\partial \mathbf{s}/\partial \mathbf{s}_x = (1,0)$, and $\partial \mathbf{s}/\partial \mathbf{s}_y = (0,1)$.

Slotted connection: The quick-return can be used to generate linear motion in $\mathbf{Z}$, at a constant value of $\mathbf{Y}$. However, such a motion changes its distance to $\mathbf{s}$ as the slider rotates. The situation is shown in the right inset. We can support such motion with a slotted connection. The slot's direction is along the slider; i.e., $\mathbf{e} - \mathbf{s}$. Its length can be determined by finding its two extreme points from the simulated motion of the attachment point. The final slider length L must be large enough to include this range of handle locations over the entire animation. In other words, the part's "handle" is not at the very end of the slider, but some distance before this, so as to always be within the slot. To simulate this connection, the IK target constrains the handle to be on the slot line. Note that the slotted connection to the quick-return differs from the ellipse cam in that we slot the part itself rather than the component it attaches to.



7 Gear

Description

In a gear system, the rotation of the driving gear around $\mathbf{X}$ forces rotation of the driven gear around $\mathbf{Y}$. The end point on a rod attached to the driven gear revolves, if placed in the center, or moves in a circle, if placed at some distance from the driven gear's center. The gear also converts one angular velocity to another.

Our system currently supports only revolute motion, not circular motion. That is, the rod must lie at the center of the driven gear. Supporting circular motion is a simple extension for future work. We also support only simple round gears, not toothed ones.

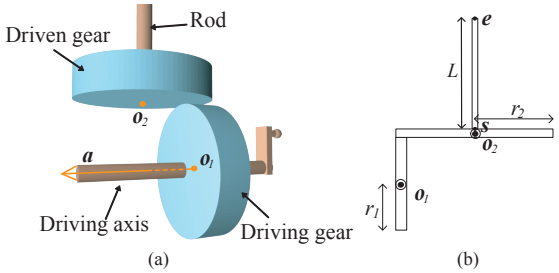


Figure 8: *Gears. (a) 3D view. (b) Schematic view.*

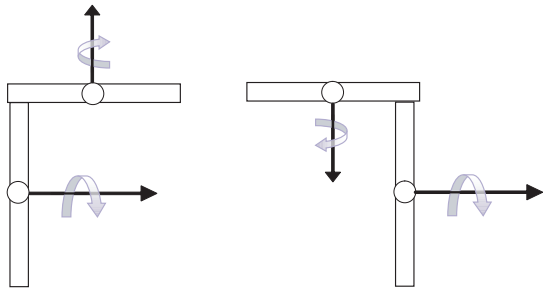


Figure 9: *Rotational sense for a gear system. The driven gear on the top rotates counterclockwise (left) or clockwise (right), as seen from above, depending on whether it is connected on the left or right of the driving gear.*

Parameters

- r_1: radius of the driving gear.
- r_2: radius of the driven gear.
- L: length of the follower rod.

Constraints

1. $r_1 < 0.5H$, where H is the height of the assembly box.

Kinematic simulation

The gear's motion is a simple rotation.

Orientation of the driving gear: The driving gear is attached to the driving axis; its angular speed is also ω. At the i-th frame, its rotation angle ωt_i.

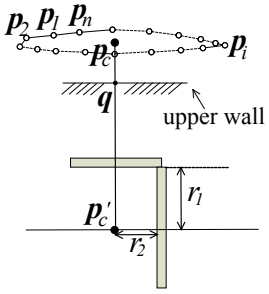
Orientation of the driven gear: The driven gear's rotation angular speed is $r_1/r_2\,\omega$. At the i-th frame, its rotation angle is $r_1/r_2\,\omega t_i$.

Initialization

If the target trajectory $\{\mathbf{p}_i, i = 1..n\}$ is circular in the canonical XZ plane, a gear system can produce that motion. We initialize it with the following steps:

1. Fit a circle to $\{\mathbf{p}_i\}$. Let $\mathbf{p}_c$ denote its center and r its radius.

2. Compute $\mathbf{p}_c'$, the projection of $\mathbf{p}_c$ onto the driving axis.



3. Compute the intersection point, $\mathbf{q}$, with the line through $\mathbf{p}_c$ and $\mathbf{p}_c'$ and the upper wall of the assembly box. Set $r_1 = 0.5\|\mathbf{p}_c' - \mathbf{q}\|$, and $r_2 = r_1\,n/m$, where m is the driving axis period and n is the period of the target motion. The driving gear must not touch the assembly box's upper wall; i.e., $r_1 < \|\mathbf{p}_c' - \mathbf{q}\|$. Otherwise, we report an error and do not initialize the gears.

4. Compute the driven gear's center $\mathbf{o}_2 = \mathbf{p}_c' + r_1\,\mathbf{Y}$. Initialize the follower rod length via $L = (\mathbf{p}_c - \mathbf{o}_2)\cdot\mathbf{Y}$.

5. Set the driving gear's center $\mathbf{o}_1$ at a distance of r_2 to $\mathbf{p}_c'$, along the driving axis. Whether this offset is left or right depends on the rotational sense of the target trajectory. See Figure 9.

Remark: Currently, the trajectory of a gear system's free end point $\mathbf{e}$ is just a fixed point. Shape parameter optimization is thus unnecessary. If the initial handle specified by the user lies along the driven gear's rotational axis, we obtain no information about its rotation. In this case we instead choose a different, nearby point on the feature component to match.

8 Belt-Pulley

Description

The belt-pulley transmits rotary motion between driving axes, as shown in Figure 10. The two axes have the same rotary direction.

Parameters

- r_1: radius of the driving pulley.
- r_2: radius of the driven pulley.
- L: distance between the two pulley centers.

Constraints

1. the two pulleys do not collide: $r_1 + r_2 < L$.

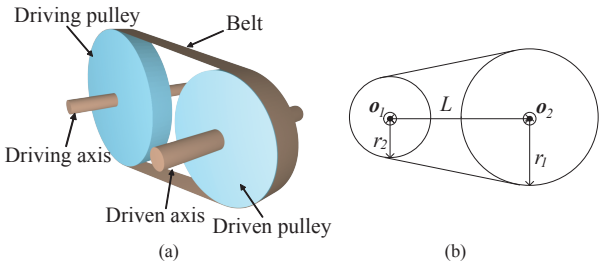


Figure 10: *Belt-pulley. (a) 3D view. (b) Schematic view.*

Kinematic simulation

The belt-pulley's kinematic behavior is similar to a gear system's.

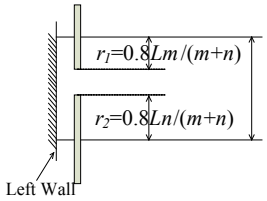
Orientation of the driving pulley: Rotation angle is ωt_i.

Orientation of the driven pulley: Rotation angle is $r_1/r_2\,\omega t_i$.

The belt slides around the two pulleys with a linear speed of ωr_1.

Initialization

1. Determine the pulley radii. For two driving axes separated by a distance L, set $r_1 = 0.8Lm/(m+n)$ and $r_2 = 0.8Ln/(m+n)$, where m is the driving axle's period and n is the driven axle's period.



2. Determine pulley positions along the driving axes. We find two corresponding positions on the two axes not occupied by any other parts in the assembly. In our implementation, we first test whether there is room close to the left or right wall of the assembly box, and if not, try the space in the middle.

References

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, SIGGRAPH '99, 317–324.