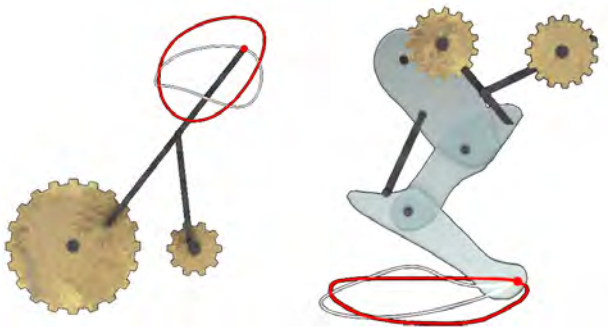

Figure 5: *The trajectory (gray polyline) traced out by a marker point (red dot) to approximate a target curve (red polyline). A marker point can either be located on the driving mechanism (*left*) or on a part of the character that is driven by a mechanism (*right*).*

old. Unfortunately, fast Poisson-disk sampling methods [Bridson 2007] cannot be applied in our setting: while it easy to satisfy the minimum distance criterion in parameter space, this is not the case in metric space since the mapping between the parameters of the mechanism and its generated motion curve is non-trivial.

We therefore generate the samples using a simple recursive process. For each accepted sample, we generate a number of new samples by uniformly probing the parameter space around the current sample's location. For each generated sample, we compute the corresponding curve, and using the distance metric described in Sec. 5, we check whether it is too close to any of the existing samples in the database. If this is the case, we reject the sample, otherwise it is added to the database. We note that it is possible that some regions in the parameter space lead to infeasible configurations. As a result, although our sampling strategy is very effective for exploring connected regions of a mechanism's parameter space, we have no guarantees that it exhaustively explores the entire space of feasible parameters.

4.2 Continuous parameter optimization

After selecting an appropriate driving mechanism from the database, its parameters need to be further fine-tuned, and we use a continuous optimization method for this purpose. We assume that a *marker* point specified on the character should follow a user-provided curve as closely as possible. The marker can either coincide with the attachment point of the driving mechanism, or it can be located at an arbitrary point on the character, as illustrated in Fig. 5. To optimize the resulting motion, we minimize the following objective as a function of the mechanisms's parameters $\mathbf{p}$:

$$F = \frac{1}{2}\int_{t=0}^{1} (\mathbf{x}(\mathbf{p},\mathbf{s}_t) - \hat{\mathbf{x}}_t)^T (\mathbf{x}(\mathbf{p},\mathbf{s}_t) - \hat{\mathbf{x}}_t) dt, \qquad (2)$$

where t is the phase of the Input Driver, such that when $t = 1$ a full cycle of the animation has been completed. $\mathbf{x}(\mathbf{p},\mathbf{s}_t)$ and $\hat{\mathbf{x}}_t$ denote the position of the marker point and its target position at phase t. The state of the assembly at phase t, $\mathbf{s}_t$, is computed by minimizing Eq. 1 after instantiating the mechanism with the current set of parameters $\mathbf{p}$. As the parameters of the mechanisms directly affect the connections between the different components, for the remainder of this section we write the system constraints as an explicit function of the parameters and the state variables i.e. $\mathbf{C}(\mathbf{p},\mathbf{s}_t)$.

Note that in Eq. (2), we use a point-based metric to measure the similarity between the input and output motion curves, rather than the more involved metric described in Sec. 5. Because the curve traced out by the marker point is already aligned with the input curve, and their shapes are relatively similar, we have found this to be sufficient. For any phase t, the gradient of the matching objective $\frac{\partial F_t}{\partial \mathbf{p}}$ is given by:

$$\frac{\partial F_t}{\partial \mathbf{p}} = (\frac{\partial \mathbf{x}}{\partial \mathbf{p}} + \frac{\partial \mathbf{x}}{\partial \mathbf{s}_t}\frac{\partial \mathbf{s}_t}{\partial \mathbf{p}})(\mathbf{x}(\mathbf{p},\mathbf{s}_t) - \hat{\mathbf{x}}_t), \qquad (3)$$

where all the partial derivatives are evaluated at $(\mathbf{p},\mathbf{s}_t)$. In the equation above, $\frac{\partial \mathbf{x}}{\partial \mathbf{p}}$ represents the change in the marker's position when the state of the sub-assembly remains unaltered. For instance, changing the location of the attachment point in the linkage structure shown in Fig 5 (left) does not affect the evolution of the state of the sub-assembly, but results in a different curve being traced out. The term $\frac{\partial \mathbf{x}}{\partial \mathbf{s}_t}$ is evaluated analytically, and the remaining term, $\frac{\partial \mathbf{s}_t}{\partial \mathbf{p}}$, represents the change in the state of the assembly as its parameters change. This term cannot be evaluated analytically because the sub-assemblies we optimize can be arbitrarily complex, and in general there is no closed-form solution describing their motion. Finite differences can be used to estimate the derivatives, but this requires Eq. (1) to be solved $O(|p|)$ times to a high degree of accuracy, which is too computationally demanding for interactive applications. Instead, we exploit the implicit relationship between the parameters of the sub-assembly and the state configuration that results in all constraints being satisfied, i.e., $C(\mathbf{p},\mathbf{s}_t(\mathbf{p})) = 0$. According to the *Implicit Function Theorem*,

$$\frac{\partial \mathbf{s}_t}{\partial \mathbf{p}} = -\frac{\partial C}{\partial \mathbf{s}_t}^{-1}\frac{\partial C}{\partial \mathbf{p}}. \qquad (4)$$

We estimate $\frac{\partial C}{\partial \mathbf{p}}$ using finite differences. This requires us to instantiate sub-assemblies with different parameters $\widetilde{\mathbf{p}}$ and compute the value of the constraints $C(\widetilde{\mathbf{p}},\mathbf{s}_t(\mathbf{p}))$. We note that this does not require Eq. (1) to be minimized, and its computational cost is therefore negligible. The matrix $\frac{\partial C}{\partial \mathbf{s}_t}$ is computed analytically. As noted in Sec. 3, our mechanical assemblies can have more constraints than degrees-of-freedom. Since the additional constraints are redundant however, the matrix $\frac{\partial C}{\partial \mathbf{s}_t}$ still has full column rank, and in such cases, we instead use the pseudo-inverse $\frac{\partial C}{\partial \mathbf{s}_t}^{+} = (\frac{\partial C}{\partial \mathbf{s}_t}^T \frac{\partial C}{\partial \mathbf{s}_t})^{-1}\frac{\partial C}{\partial \mathbf{s}_t}^T$.

We minimize Eq. (2) by evaluating the matching objective and its gradient at a discrete number of points (i.e., t values). To improve convergence rates we use the BFGS quasi-Newton method to estimate the approximate Hessian $\frac{\partial}{\partial \mathbf{p}}\frac{\partial F}{\partial \mathbf{p}}$ [Nocedal and Wright 2006].

4.3 Timing control via non-circular gears

Controlling the timing of motions is an integral part of character animation [Lasseter 1987]. We therefore also allow users to explicitly control the timing of the characters' motions. This is accomplished through the use of non-circular gears, which are always used in pairs: a constant angular velocity input signal applied to the first gear gets transformed to variable angular velocity for the second gear. The phase-dependent ratio of angular velocities of the two gears depends on the shape of their profiles, and in particular, on the ratio of the gears' radii at the point where they mesh. Asking users to directly provide the phase-dependent ratio of angular velocities, however, is not intuitive, as the integral of this ratio needs to remain constant (otherwise one gear rotates more than the other over a full cycle). Instead, we allow users to define the relative phase profile between the two gears, as illustrated in Fig 6 (left). This is accomplished by allowing the user to set samples $(\alpha_{1i}, \alpha_{2i})$