The design of a compact, two-part torsion spring architecture

Zachary Bons[1,2†], Gray C. Thomas[1,3†], Luke Mooney[4], Elliott Rouse[1,2,3*]

[1]Neurobionics Lab, University of Michigan, Ann Arbor, USA
[2]Department of Mechanical Engineering, University of Michigan, Ann Arbor, USA
[3]Department of Robotics, University of Michigan, Ann Arbor, USA
[4]Dephy Inc., Maynard, MA, USA

[*]To whom correspondence should be addressed; E-mail: ejrouse@umich.edu.
[†]Co-first authors

Emerging wearable and assistive robots, such as the Open-Source Leg (OSL), seek to interact with the environment and/or humans in a compliant, dynamic, and adaptable way. Springs are critical to achieving this objective, but the associated increase in volume, mass, and complexity is limiting their application and impact in this rapidly developing field. This paper presents a novel rotary spring architecture that is both light and compact. In order to facilitate adoption, we introduce an open-source design tool which enables the design of custom springs within a matter of minutes. Four test springs validate the achievement of target spring rates and deflections in an automated dynamometry testbed.

Introduction

Springs are essential building blocks for many mechanical engineering applications, including storing elastic energy and measuring force or torque. This combination of functionality makes springs highly desirable in human-centered robotic applications (*e.g.*, rehabilitation devices, exoskeletons, and prostheses), but the added weight and complexity of incorporating springs can result in a difficult trade-off for these mobile devices (*1–5*). One particularly common use-case in these robots involves implementing a spring between the output of the transmission and the load, and is known as a Series Elastic Actuator (SEA) (*6*). Though not without drawbacks—namely force bandwidth reduction and the aforementioned increase in mass and complexity—this design paradigm has notable benefits, including compliant interaction with the environment, torque feedback, energy storage, and improved shock tolerance. In recent years, the SEA application has become a driving force in torsion spring designs. For mobile robot systems, these designs have prioritized specific energy (energy storage per mass) and energy density (energy storage per volume).

As SEA systems become more prevalent, simple torsion spring designs, including thin-walled tubes and cantilever beams, have posed challenges in packaging and volume. Torsion tube springs with thin walls (*7, 8*) and long beam flexures with cam-rollers (*9*) or hinges (*10*) can be effective for low stiffness applications, and their low weight results in high specific energy. However, to achieve higher stiffness values, the aspect ratio—length over diameter—must be considerable (*11*). Thus, tubular geometries are often impractical for rotary joints in robots. This challenge has resulted in an era of spring designs that prioritize convenient packaging in addition to specific energy and energy density.

To achieve convenient packaging, recent designs have emphasized disk-like architectures that balance specific energy, compactness, and required mechanics. These springs incorporate

torsional elasticity between a central anchor point rotated with respect to an outer rim. Some early researchers achieved this design by arranging prismatic springs tangentially around a circular frame (*12, 13*). However, this approach results in a nonlinear stiffness behavior (*13*) and has the added complexity of several removable parts. The latter concern may be responsible for the transition to monolithic springs in subsequent designs. One of the most common and successful planar monolith spring designs connects the central anchor to the rim with one or more spiral arms (*14–19*). This configuration is efficient and works well with a wide range of stiffness coefficients (*e.g.*, 30-800 Nm/rad); however, the nature of the spiral arms often causes differences in stiffness depending on the direction of deflection (*19*). Other designers have instead emphasized torque-sensing resolution (*5, 20–25*) in their development. These designs have produced compact springs that are generally limited in their deflection (*e.g.*, less than five degrees) and torque limits, which has diminished their success in many SEA applications. While the overall spring performance has been well-predicted in some cases (*17, 18*), the directionality and limited range of motion are drawbacks of these approaches.

One method to improve the energy density of springs is to modify the loading condition of the elastic elements within the spring. That is, the above-mentioned monolith spring designs fully constrain both ends of the spring which prevents the material from achieving ideal bending thus leading to lower energy density and specific energy of the final spring. In addition, they are constrained by the fact that strain rate must be continuous as a function of spring radius which ultimately limits potential deflection. One way to shift towards more efficient loading conditions and eliminate the strain constraint is to develop springs that can be assembled from multiple parts. For example, Herodotou *et al.* proposed a novel design in which one end of the arm is fixed and the other end is hinged (*26*). This example is quite efficient (by mass) and the stiffness was accurately predicted by models and finite element analysis (FEA). Drawbacks of these designs are that they add complexity, and have thus far only been developed for extremely

high stiffness values (1950 Nm/rad). However, it is possible that lower stiffness behaviors could be achieved with a similar strategy.

A spring design that fully exploits an efficient loading condition should be capable of achieving low stiffness behaviors while also maintaining a compact form-factor and low mass. In addition, a design that employs a convenient mate with transmission components (*e.g.*, fitting inside the timing belt pulley of a belt-drive transmission (*27*)) would be novel and desirable. Inspired by (*28*), we developed a two-element planar spring design that addresses these gaps, is predictable and customizable, and has high energy-per-mass and energy-per-volume efficiency. The preliminary design and findings were presented previously (*29*), but significant developments will be shared herein.

In this paper we 1) introduce a compact and lightweight rotary spring design based on simple design indices, 2) present a design tool for generating spring profiles from the design indices, and 3) empirically validate four representative test springs' designs. These contributions enable rapid adoption of an energy-efficient spring design and thus pave the way for widespread inclusion of springs in lightweight and compact technologies like wearable robots.

Results

Spring Architecture

In this section, we present the design of our spring topology (Fig. 1) and provide the governing equations that describe the mechanics and geometry of the individual flexures. Our intent is to motivate the rationale for our design decisions as well as provide a guide for other researchers to modify for other applications. In addition, we have developed a spring design tool and graphical user interface (see Methods), which is openly-available in [TBD].

Our spring design (Fig. 2a) comprises a gear-like camshaft in contact with a ring of radial flexure teeth. The teeth protrude inward from a shared outer ring, which is fixed to the spring

housing. In operation, the gear-like camshaft rotates relative to the rim ring, imposing a contact force upon the tip of each tooth and causing each of the inward-pointing flexures to bend like a cantilever beam. The rim of the spring is fixed in place using dowel rods that oppose rotation between the spring and the inner bore of the housing (Fig. 1a). This is accomplished by designing both the rim ring and the spring housing with complementary semi-circular cutouts, and inserting the dowel rods in the hole when the spring and housing are properly aligned.

Camshaft Design

The designed loading condition of this spring enables maximum strain—and thus energy storage—of the bending flexures. Typical monolith springs are limited in this respect due to the constraint that the strain rate must be continuous as a function of spring radius. The relative motion that occurs between the camshaft and spring in our two-part design eliminates this constraint. Taking advantage of this extra degree of freedom, we specifically designed the cam-spring interface to approximate ideal bending: the most efficient loading condition for bending beams.

Ideal bending occurs when a pure moment is imposed along the full length of a beam. This condition can be approximated by applying a force to the free end of the beam, with the force being perpendicular to the neutral axis of the beam. Under this condition, the stress due to the force is small compared to the stress caused by the induced moment and it therefore resembles a pure moment.

To achieve an applied force that is nominally perpendicular to the length of each flexure, we designed the contact interface to be a circular flexure tip that slides along an involute cam profile. The derivation of the cam profile was accomplished by approximating the path of the flexure tip as parallel to the y-axis (see Fig. 2b) throughout the range of spring deflection (*i.e.*, small angle approximation) and iteratively resolved the necessary geometric constraints: at each position along the approximated tip path the camshaft must be tangent to the circular tip and

the contact point must be coincident with the line of the tip path. These constraints resulted in a cam profile that is involute to the contact radius and ensures the prescribed perpendicular contact force.

Straight Tapered Flexures

To maximize the specific energy of the spring, each inward-pointing flexure is tapered. The tapering law (*30*) ensures that the entirety of the two bending surfaces (Fig. 2b) reach the desired design stress at peak deflection. Consequently, a significant amount of material can be removed from the standard non-tapered beam, increasing the ratio of energy storage to mass. This tapering law governs λ (see Fig. 2b) as a function of x (the distance along each flexure), and is derived from basic beam-bending mechanics (*31*).

For a generic beam, we know $\sigma = \frac{M\lambda}{I}$, with σ as axial stress, M as applied moment ($F(L-x)$ for our loading condition), λ as the distance from the neutral axis to the edge of the beam (see Fig. 2b), and I as the second moment of area. In the case of a planar spring, we have $I = \frac{2t\lambda^3}{3}$, with t as the thickness of the spring (see Fig. 2b). To achieve maximum stress along the length of the beam, we choose λ at each cross-section such that $\sigma = \sigma_d$:

$$\sigma_d = \frac{3F(L-x)\lambda}{2t\lambda^3}, \tag{1}$$

$$\lambda(x) = \sqrt{\frac{3F(L-x)}{2t\sigma_d}}. \tag{2}$$

This tapering law fully constrains the flexure geometry and can be used to relate spring rate and bending strain energy. By equating bending strain energy and the desired energy storage of the spring, we can then predict deflection properties of the spring.

Assuming that the flexure-cam interface indeed approximates ideal bending, we can easily calculate strain energy. For a generic beam in bending, strain energy (U) is defined by $U = \frac{M\theta}{2}$, with M as previously defined and θ as deflection angle. We also know that $\theta = \kappa L$, where

(curvature) $\kappa = \frac{M}{EI}$. Thus, we see that $U = \frac{M^2 L}{2EI}$. For a varying beam profile with small deflections, we can rewrite strain energy as follows:

$$U = \int_0^L \frac{F^2(L-x)^2}{2EI}\, dx. \tag{3}$$

Substituting our definitions of I and λ yields

$$U = \int_0^L \frac{3F^2(L-x)^2}{4Et\sqrt{\frac{3F(L-x)}{2t\sigma_d}}^3}\, dx. \tag{4}$$

Rearranging and simplifying,

$$U = \frac{\sigma_d^2 t}{3E} \int_0^L \sqrt{\frac{3F(L-x)}{2t\sigma_d}}\, dx, \tag{5}$$

which can be rewritten in closed form as

$$U = \sqrt{\frac{2tFL^3\sigma_d^3}{27E^2}}. \tag{6}$$

Since force (F) is a function of stiffness (k), desired deflection (θ_{des}), the number of flexures (n) and the flexure-camshaft contact radius (r), this can be further simplified to

$$U = \sqrt{\frac{2tk\theta_{des}L^3\sigma_d^3}{27E^2rn}}. \tag{7}$$

The desired energy storage of a single flexure can be calculated by simply dividing the elastic potential of the full spring by the number of flexures:

$$\mathcal{E} = \frac{1}{2n} k\theta_{des}^2. \tag{8}$$

Therefore, equating the two expressions for energy storage within a flexure–(7) and (8)–and solving for θ_{des} allows us to predict the spring deflection as a function of spring design variables,

$$\theta_{des} = \sqrt[3]{\frac{8tnL^3\sigma_d^3}{27E^2kr}}. \tag{9}$$

Thus, while the tapering law characterizes mass-efficient straight flexures, specific stiffness (k), geometry (r, L, n, t), and material (E, σ_d) constraints directly limit the possible deflection of the spring (9). In addition, the energy density of the spring disks is limited due to the necessary gaps between flexures to accommodate deflection and the gear-like cam shaft.

Serpentine Tapered Flexures

To maximize energy density while maintaining high specific energy, it is possible to design *serpentine* flexures that follow the tapering law. This results in beams that are effectively longer than the *straight* flexures, yielding higher energy storage through increased flexure deflection for the same outer diameter. A serpentine design also makes better use of the gaps between flexures, resulting in higher overall volume-efficiency in comparison to the straight flexure designs.

To parameterize the design of serpentine flexures, we first rewrite the expression for strain energy (Eq. 5) by substituting our definition of λ (Eq. 2),

$$U = \frac{1}{6}\frac{\sigma_d^2}{E} t (2 \int_0^L \lambda \, dx), \tag{10}$$

which can be further generalized to

$$U = \frac{1}{6}\frac{\sigma_d^2}{E} t A. \tag{11}$$

Energy storage is therefore a function of the planar area of the flexures (A), so by appropriately increasing A, we can increase energy storage. Equating desired energy storage (8) with our updated expression for strain energy (11), we easily determine the required planar area (A_{serp}) of a flexure that achieves our desired peak deflection and torque,

$$A_{\text{serp}} = \frac{3k\theta_{des}^2 E}{n\sigma_d^2 t}. \tag{12}$$

Therefore, knowing the required planar area (A_{serp}) and the governing tapered profile, one can define a serpentine flexure that satisfies the constraints. With this approach, spring deflection and stiffness can be prescribed independently, yielding increased design flexibility. To aid in the classification of a spring with such flexures, we introduce two design indices that indicate feasibility of a spring:

- **Serpentine factor (f_s)**: The serpentine factor is calculated as $f_s = \frac{A_{\text{serp}}}{A_{\text{nom}}}$, where A_{nom} is the planar area of the straight flexures—and describes the sinuosity of a given flexure.

If $f_s = 1$, then the desired spring will have straight flexures. If $f_s < 1$, the flexure is undefined, and the diameter of the spring can be decreased while still maintaining the required deflection. Lastly, if $f_s > 1$ flexures should have a serpentine shape in order to achieve desired performance.

- **Density factor (f_d)**: A density factor is calculated as $f_d = \frac{A_{\text{serp}} n}{A_{\text{annulus}}}$, where n is the number of flexures in the spring and A_{annulus} is the annular (donut-shaped) area in which the flexure teeth lie. This indicator describes the compactness of the flexures within the the spring: a value of 1 would indicate that the spring is a solid disk (all flexures are touching) whereas a value of 0 would represent a spring with no flexure teeth. In the testing performed, we have had difficulty designing springs that don't self-intersect with $f_d > 0.55$.

Additional Factors

Several other design parameters have large effects on energy-storage potential, namely the number of flexures and the flexure-camshaft contact radius. First, increasing the number of flexures within the spring reduces the empty space between flexures—similar to the serpentine concept—and thus increases energy density. Second, decreasing the contact radius makes space for longer flexures, and therefore also increases the energy density of the spring.

The direct relationship between these parameters and spring deflection can be seen by returning to the expression for desired spring deflection (9). Thus, even when all geometry and material parameters are fixed, increasing the number of flexures (n) further increases deflection and therefore energy storage. Similarly, decreasing contact radius (r) directly improves deflection, and it also increases L ($L = R - r$) which again increases spring deflection. Practical limitations typically govern the possible number of flexures and a feasible contact radius (*e.g.*, nearness constraints of flexure tips). Therefore, proper selection of these parameters can greatly

enhance spring performance.

To summarize, by designing springs with *ideal bending* of *tapered* flexures, we can achieve high specific energy. Selecting the *optimal number* of flexures and imposing a *serpentine* geometry increases the energy density.

Hardware Validation

We empirically validated our design framework by comparing four springs with differing geometries and identical desired stiffness coefficients (Fig. 1 d-g). The four springs intentionally employ different serpentine factors and root radii (*i.e.*, to simulate springs with a smaller outer ring diameter) to highlight the impact of the flexure geometry on spring stiffness and energy storage capacity. Specifically, the springs tested in this study were as follows (Table 1):

- **Spring One** (S1) uses 24 straight flexures with a 31 mm root radius, and serves as a baseline comparison for the other designs.

- **Spring Two** (S2) uses 24 flexures with a moderate serpentine factor, 1.24, to demonstrate the increased allowable deflection of serpentine flexures when compared to the straight flexures of S1. Its flexures were designed with a 31 mm root radius—equivalent to that of S1.

- **Spring Three** (S3) illustrates how serpentine flexures can enable similar performance to the straight flexures of S1 within a smaller enclosed volume. It was also designed with 24 flexures, but with an aggressive serpentine factor of 1.32 and a 26 mm root radius—substantially smaller than that of S1 and S2.

- **Spring Four** (S4) was designed to demonstrate the combined effect of using serpentine flexures after optimizing the number of flexures and the flexure-camshaft contact radius

for maximum deflection. It has a light serpentine factor of 1.17, 31 flexures, a 5.1 mm contact radius, and the same 26 mm root radius as S3.

All four springs were manufactured from hardened SS420, and were designed with a target spring rate of 150 Nm/rad, spring thickness of 4.5 mm, and identical outer radius (33.5 mm) to interface with the spring housing.

The measured performance of the springs closely matched the desired specifications. First, each spring achieves its designed deflection limits without signs of failure (Fig. 3). In addition, the measured spring rates closely align with the target spring rate (Table 1), falling within 6% of the intended value for all four springs (Table 2). We also quantified energy loss due to hysteresis during the loading and unloading process. At designed deflection (as depicted) the percent energy loss is approximately 5% for the first three springs, but higher for the fourth (S1: 3.77%, S2: 5.36%, S3: 4.82%, and S4: 13.01%). However, at smaller deflections of 0.16 radians, those percentages are significantly lower (S1: 2.30%, S2: 2.40%, S3: 0.84%, and S4: 2.09%).

Though not empirically validated, we also designed our springs to last roughly 100,000 alternating cycles under full load. Using S-N curve estimation as outlined in (*32*), we determined the design stress (912 MPa) that achieves 100,000 cycles with our material (SS 420). This stress limit was used as an input to the design tool when creating all four springs.

Discussion

Contributions

In this paper, we introduced a novel two-part spring architecture, along with an open-source design tool that enables rapid customization and implementation of the spring. In addition, we presented an empirical validation of four representative springs to demonstrate the correspon-

dence between our mechanical model and the physical realizations. The intent of our spring design was to fit within the volume of common transmission components used in the Open-Source Leg (*27*) and other robotic systems, thus enabling series elasticity of the joints without increasing their size.

This design objective led to a two-part spring architecture comprised of a ring of radially-spaced cantilever flexures in contact with an involute camshaft that lies in the center of the ring. We intentionally made several decisions in the design process to maximize energy storage and density. First, the two-part design allows relative motion between the camshaft and the spring, which removes the constraint of a continuous strain rate as a function of spring radius. Second, a tapered profile induces equivalent stress along the full length of the flexures, thus eliminating unnecessary mass. Third, serpentine flexures allow greater deflection and length—and thus energy storage—when compared to straight flexures of the same spring radius. To our knowledge, these innovations result in a new class of torsion springs with the highest specific energy and energy density to date (Fig. 4).

We achieved high agreement between our mechanical model and our empirical results. The mechanical model of the spring mechanics involved several simplifying assumptions, including the forces being perpendicular to the nominal flexure, ideal bending along the flexure, and small angles of deflection. The low error between the empirically-determined and desired stiffness coefficients (1-6%) indicates these assumptions had minimal effect in the torsion spring mechanics. Furthermore, the springs achieved their respective desired deflections and the observed backlash was minimal despite the two-part design (Fig. 3a, b). We also accounted for fatigue life by appropriately modifying the design stress, according to an estimated S-N curve (*32*). All four springs were designed to last over 100,000 loading cycles.

To facilitate the implementation of springs within mechanical and robotic systems, we developed an open-source design tool that outputs custom spring designs—according to the user's

inputs—in a matter of minutes [REF]. The tool couples an optimization routine with our mechanical model to automatically design the spring and camshaft that satisfies the design requirements. The spring geometry can then be exported for direct implementation in solid modeling software for design. Future researchers that wish to include our springs in their designs can now do so without the overhead of mathematical derivations or iterative finite-element analysis.

Broader Applicability

This spring design is highly customizable and easily implemented, making it suitable for a broad range of applications. Immediate areas of impact will likely include series and parallel elastic actuators, and other mechanisms in which torsional compliance is desirable. The unique combination of compactness and high energy storage not only make this spring particularly useful in systems that prioritize low mass and volume (*e.g*, wearable robotic systems like exoskeletons and robotic prostheses), but also in serial-link manipulators, humanoids, and other mobile robots where mass and volume are design-driving factors. As an example of the potential impact of our approach, we compared our spring architecture to five existing spring designs used in powered mobile robots (Fig. 5). We used our open-source tool to design springs that matched the stiffness and deflection requirements of the springs in a lower-limb rehabilitation exoskeleton (*16*), an upper-limb rehabilitation exoskeleton (*15*), a lower-limb robotic orthosis (*5*), a planar bipedal robot (*4*) and the humanoid Robonaut 2 (*33*). In every case, the output of our design tool resulted in a spring that was both lighter and more compact than the original spring (Table 3). Notably, the process of designing these spring examples with the open-source design tool took less than 10 minutes from knowing the design requirements to completing the solid model.

Further modifications to the spring design could increase applicability. For example, implementing series connections of springs would yield even greater deflections (*20*), or capitalizing

on the underconstrained interface between the cam and the spring could result in a sliding clutch or variable stiffness mechanism. In addition, the bending beam paradigm could be directly applied prismatic springs, which could be useful as high-efficiency alternative to springs in series with linear actuators, or in mechanisms such as suspension systems.

Limitations

Our design framework produces compact springs that match user specifications, but future work could improve or further validate the approach. First, the sources of hysteresis could be investigated in depth. It is likely that the hysteresis is due in large part to the sliding contact between the camshaft and the spring; however, the friction could be significantly reduced by lubricating the contact surfaces or improving their surface finish. Interestingly, the fourth test spring (S4) had much larger energy loss due to hysteresis than the other three springs (13% compared to $\sim$5%). The major differences in the design of this spring include a larger number of flexures (30% increase), a smaller contact radius (15% decrease) and a smaller flexure-tip diameter (33% decrease). Thus, one plausible explanation for the difference in hysteresis could stem from the increased friction due to a larger number of contact surfaces.

Second, the slope of the measured torque-deflection curves is not perfectly constant, and it generally increases as the springs approach their deflection limits (Fig. 3a). While the spring rate is likely sufficiently linear for the majority of spring applications, there are certain situations in which high precision is needed (*e.g.*...). For such situations, the linearity could potentially be improved by modifying the shape of the flexure tip and/or the camshaft.

Conclusion

In this paper, we presented a novel spring design characterized by a ring of radially-extending tapered cantilever beams in contact with a gear-like camshaft. This new paradigm yields the

most energy-efficient springs to date. In addition, we introduced a design tool that will facilitate rapid adoption and customization of this technology. We also tested four representative springs on a custom testbed to demonstrate the high fidelity of our design framework. This work increases the potential for incorporating springs into compact mechanical systems.

Materials and Methods

Experimental Protocol

The purpose of our hardware testing was to empirically validate our spring design approach where each spring was designed with identical stiffness values. Our experimental apparatus (Fig. 6) deflected the spring by actively using two opposing actuators, while simultaneously measuring the deflection and torque at the spring interface. The springs were held in a housing with complementary semi-circular cutouts on the inner bore (see Section II). The applied moments were opposed by inserting dowel rods in the circular cutouts at eight evenly-spaced locations around the spring. Torque was measured with a contactless sensor (TRS605, Futek, Irvine CA) in series with the spring assembly. The torque sensor output an analog voltage that was sampled by an 16-bit analog-to-digital converter at 265 Hz. Rotary motion was provided by two identical brushless DC actuators (ActPack, Dephy Inc, Maynard MA), each coupled to a transmission (50:1 PL2090-050, Boston Gear, Boston MA), as used in (*34*). The actuators were controlled in current or position control modes by a microprocessor (RPi 3 Model B+, Raspberry Pi Foundation, Cambridge UK). Both actuators include an encoder, however, deflection within the transmission and testbed setup added error therefore we had to obtain the spring deflection separately. Spring deflection was directly measured using a custom optical encoder and image processing algorithm. Two arrays of optical fiducials were used for the measurement: one set was rigidly mounted to the shaft that engaged the spring, while the other set was fixed to the spring housing. Tracking these two fiducial arrays enabled deflection to be sensed locally

at the spring interface, improving the quality of our measurements.

A dedicated high-definition camera provided input to our image processing algorithm. The camera (RPi Camera Module 2, Raspberry Pi Foundation, Cambridge UK) recorded the movement of the optical fiducials during motion and performed an off-line analysis to determine relative angular displacement (Fig. 7). We used filter masks to reduce the image to the areas of interest (the two arcs containing fiducials). Subsequently, we used OpenCV (*35*) to track the fiducials in the plane of the image, and determined the displacement using best-fit ellipses obtained from a calibration procedure. The accuracy of this approach was validated by comparing the measured displacement to that measured by the motor encoders during unload, and the resulting error fell within 95% CI [0.0442, 0.0443] radians. To improve the measurement, we used the relationship between true angle and camera-measured angle (obtained with a no-load calibration procedure) to remove periodic nonlinearities due to optical effects from camera-measured angles. With this additional step, the error of the corrected camera-measured angles reduced to 95% CI [0.00079, 0.00082] radians.

The testing that we performed thoroughly spanned the expected operating ranges of the springs. With each spring, we increased the deflection in one direction to a specified limit, then subsequently decreased the deflection to zero and repeated the process in the opposite direction. We began with two degrees of deflection in both directions and increased the allowable deflection in 1-2 degree steps until we reached deflections well past the designed limit. Each ramp lasted five seconds, so smaller angles of deflection also had a slower associated angular velocity. By comparing deflection and torque measurements, we quantified the torque-angle relationship of each spring and the maximum ranges of safe operation.

Flexure Profile Optimization

Since serpentine geometries have many degrees of freedom, we use optimization to automatically define the flexures and offload the burden of geometry selection from the user. Using our software, we can quickly generate one of the many spring profiles that meets desired specifications. Our approach is based on nonlinear optimization that is constrained to achieve the target planar flexure area (among other requirements). We developed this tool to enable quick and simple adoption of this new spring design to lower the barrier to entry for devices with custom elastic elements (scripts and instructions available @TBD). The published library includes more specific documentation for its use in the MATLAB language, and generates output files that can be loaded into complete computer aided design packages like SolidWorks as a 2D sketch component from which the solid body of the spring can be extruded.

The primary objective of our profile optimization is to design a spring flexure that has the desired serpentine factor (f_s) and thickness profile, thereby meeting the performance requirements of the spring. The spring designs are parameterized in such a way that the planar area (and thus the serpentine factor, stiffness, and total stored energy) can be adjusted by the constrained nonlinear optimization without increasing the packaging volume of the spring. Since the serpentine geometry has many degrees of freedom, the optimization is under-defined without secondary objectives. Thus, the optimization includes constraints that align with the assumptions made during the derivation of the design. Specifically, these constraints impose flexure configurations with straight tips, straight roots, laterally balanced geometry, smooth curves, and low likelihood of self-collision in addition to the planar area requirement.

We parameterized the shape of our flexure using the center curve and thickness profile—both being inputs to the optimization. The center curve is defined using a cubic spline interpolation with end conditions. Splines were chosen for their robust manipulability. The end conditions pre-define the desired zero-slope conditions (see XY-plane in Fig. 2b) at both the tip and the root

of the flexure, while the interior control points allow precise manipulation of the shape between the two ends. During optimization, the number of control points is held constant, as well as the overall packaging format of the spring (*e.g.*, outer radius of the flexure disk and nominal radius of the gear-like camshaft). Points (x_{edge}, y_{edge}) along the boundary of the flexure area are technically defined by moving a distance of λ (Eq. 2) in the direction perpendicular to the slope (m) of the spline at each point (x_c, y_c) along the center curve:

$$\begin{bmatrix} x_{\text{edge}} \\ y_{\text{edge}} \end{bmatrix} = \begin{bmatrix} x_c \\ y_c \end{bmatrix} \pm \begin{bmatrix} 1 \\ -m \end{bmatrix} sgn(m) \frac{\lambda(x_c)}{\sqrt{m^2+1}}. \tag{13}$$

Area is computed numerically using a high-resolution approximation of the flexure curve.

Due to the presence of many local minima in our objective function (which complicates convergence to a global optimum), we include run-time as a user-defined input, and restart the optimizer repeatedly from randomized initial conditions until time expires. The optimization is implemented using MATLAB's fmincon, and the large number of computations cause it to take several seconds to solve one set of initial conditions. We tested the implementation on a 4-core processor @ 1.50 GHz and we have found that setting the run time to 30 seconds allows the tool to find several viable spring designs. The final spring design is then automatically selected from the solutions of all trials.

Several constraints ensure a feasible solution of the nonlinear optimization. These constraints use a 2D coordinate system with x pointing outward from the center of the spring disk along the nominal center of the flexure.

- The x-coordinates of the n interior control points are constrained to be in descending order.
- The corresponding y-coordinates are constrained to the range $[-\frac{L}{2}, 0]$ or $[0, \frac{L}{2}]$ for odd or even points, respectively. This forces the spline to be roughly centered about the x-axis.

- The flexure face is constrained to have exactly the desired planar area. As shown in the characterization of the spring design, this is necessary to achieve desired spring performance. It is implemented by limiting the difference between the desired and actual serpentine factors ($f_{s,des}$, f_s) to less than 0.001.

- The spring is also constrained to be laterally balanced across the x-axis, which ensures that the spring will have similar performance in both loading directions. This is implemented as follows, where n_c is the number of points along the centerline spline and $y_{c,i}$ is the ith y-coordinate along that spline:

$$\left| \sum_{i=1}^{n_c} y_{c,i} \right| < 0.001. \tag{14}$$

The cost function is defined to penalize a combination of the sharpness of curvature and nearness to self-collision between flexures. Eliminating sharp curves is necessary to avoid the nonlinearities associated with stress concentrations. This is achieved by penalizing high curvature along the centerline spline. Curvature is calculated numerically and the sum of the squares of the curvature is included in the cost,

$$c_{\text{sharpness}} = \left| \sum_{i=1}^{n_p} K_i^2 \right|. \tag{15}$$

Collision of neighboring flexures leads to premature failure of the spring, so the best spring will have a large minimum distance (d_{min}) between neighboring flexures. The minimum distance is calculated by checking the distance between every point along neighboring curves. The minimum is then stored, and its inverse is added to the cost, making small distances expensive:

$$c_{\text{nearness}} = \frac{1}{d_{min}}. \tag{16}$$

Total cost is then calculated as a weighted sum of $c_{\text{sharpness}}$ and c_{nearness}.

To facilitate dissemination of our spring concept, we developed an open-source design tool that implements this profile optimization to automatically design a matching spring and

camshaft according to customizable user inputs. Inputs include basic geometric constraints, material properties and performance specifications. The 2D spring profile is output as xyz points in a .txt file, which can be easily imported to CAD software and extruded to achieve the full 3D model. We utilized this tool to design the springs used in the experimental validation of this spring design.

References

1. J. Pratt, B. Krupp, C. Morse, *Industrial Robot* **29**, 234 (2002).

2. C. Fitzgerald, *Technologies for Practical Robot Applications (TePRA), 2013 IEEE Conference on* (IEEE, 2013), pp. 1–6.

3. K. Kong, J. Bae, M. Tomizuka, *IEEE/ASME Transactions on Mechatronics* **14**, 105 (2009).

4. T. Kamioka, H. Shin, R. Yamaguchi, M. Muromachi, *2022 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2022).

5. D. Accoto, *et al.*, *International Journal of Advanced Robotic Systems* **10**, 359 (2013).

6. G. A. Pratt, M. M. Williamson, *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on* (IEEE, 1995), vol. 1, pp. 399–406.

7. M. M. Williamson, Robot arm control exploiting natural dynamics, Ph.D. thesis, Massachusetts Institute of Technology (1999).

8. G. A. Pratt, *Integrative and Comparative Biology* **42**, 174 (2002).

9. M. K. Shepherd, E. J. Rouse, *Transactions on Neural Systems and Rehabilitation Engineering* **25**, 2375 (2017).

10. C. Hubicki, *et al.*, *The International Journal of Robotics Research* **35**, 1497–1521 (2016).

11. F. Negrello, *et al.*, *2017 IEEE international conference on advanced intelligent mechatronics (AIM)* (IEEE, 2017), pp. 271–278.

12. S. Yoon, *et al.*, *2003 IEEE/RSJ international conference on intelligent robots and systems* (IEEE, 2003), pp. 2191–2196.

13. N. G. Tsagarakis, M. Laffranchi, B. Vanderborght, D. G. Caldwell, *2009 IEEE international conference on robotics and automation* (IEEE, 2009), pp. 4356–4362.

14. B. T. Knox, J. P. Schmiedeler, *Journal of Mechanical Design* **131** (2009). 125001.

15. A. H. Stienen, *et al.*, *Transactions on Biomedical Engineering* **57**, 728 (2010).

16. C. Lagoda, A. C. Schouten, A. H. Stienen, E. E. Hekman, H. van der Kooij, *2010 3rd IEEE RAS & EMBS international conference on biomedical robotics and biomechatronics* (IEEE, 2010), pp. 21–26.

17. S. Wang, C. Meijneke, H. van der Kooij, *2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR)* (IEEE, 2013), pp. 1–8.

18. N. Georgiev, J. Burdick, *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2017), pp. 4777–4784.

19. S. Yoo, J. Lee, J. Choi, G. Chung, W. K. Chung, *Mechatronics* **43**, 112 (2017).

20. G. Carpino, D. Accoto, F. Sergi, N. Luigi Tagliamonte, E. Guglielmelli, *Journal of Mechanical Design* **134** (2012). 121002.

21. N. Paine, *et al.*, *Journal of Field Robotics* **32**, 378 (2015).

22. W. M. dos Santos, G. A. Caurin, A. A. Siqueira, *Control Engineering Practice* **58**, 307 (2017).

23. T. Kim, K. Shi, K. Kong, *2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)* (IEEE/ASME, 2021), pp. 572–577.

24. G. G. Fiorezi, J. dos Santos de Moraes, P. H. Fabriz Ulhoa, R. M. de Andrade, *Proc. First International Electronic Conference on Actuator Technology* (MDPI, 2020), pp. 1–11.

25. J. P. Cummings, *et al.*, *Journal of Mechanisms and Robotics* **8**, 1 (2016).

26. P. Herodotou, S. Wang, *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)* (IEEE, 2019), pp. 331–336.

27. A. F. Azocar, *et al.*, *Nature Biomedical Engineering* **4**, 941 (2020).

28. L. Mooney, K. A. Pasch, T. D. Doan, Planar torsion spring for knee prostheses and exoskeletons (2017). US Patent App. 15/402,186.

29. Z. Bons, G. C. Thomas, L. M. Mooney, E. J. Rouse, *Robotics and Automation (ICRA), 2023 IEEE International Conference on* (IEEE, 2023), pp. –.

30. M. K. Shepherd, E. J. Rouse, *TRANSACTIONS ON MECHATRONICS* **22**, 1695 (2017).

31. B. J. Goodno, J. M. Gere, *Mechanics of Materials, Ninth Edition* (Cengage Learning, 2013).

32. J. A. Collins, H. Busby, G. Staab, *Mechanical Design of Machine Elements and Machines, Second Edition* (Wiley, 2010).

33. M. A. Diftler, *et al.*, *IEEE International Conference on Robotics and Automation (ICRA)* (2011), pp. 2178–2183.

34. E. A. Bolívar-Nieto, G. C. Thomas, E. Rouse, R. D. Gregg, *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2021), pp. 9327–9332.

35. G. Bradski, *Dr. Dobb's Journal of Software Tools* (2000).

36. G. Zhao, M. A. Sharbafi, M. Vlutters, E. van Asseldon, A. Seyfarth, *Transactions on Neural Systems and Rehabilitation Engineering* **27**, 1760 (2019).

Acknowledgments

Include acknowledgments of funding, any patents pending, where raw data for the paper are deposited, etc.

Supplementary materials

Materials and Methods

Supplementary Text

Figs. S1 to S3

Tables S1 to S4

References *(4-10)*

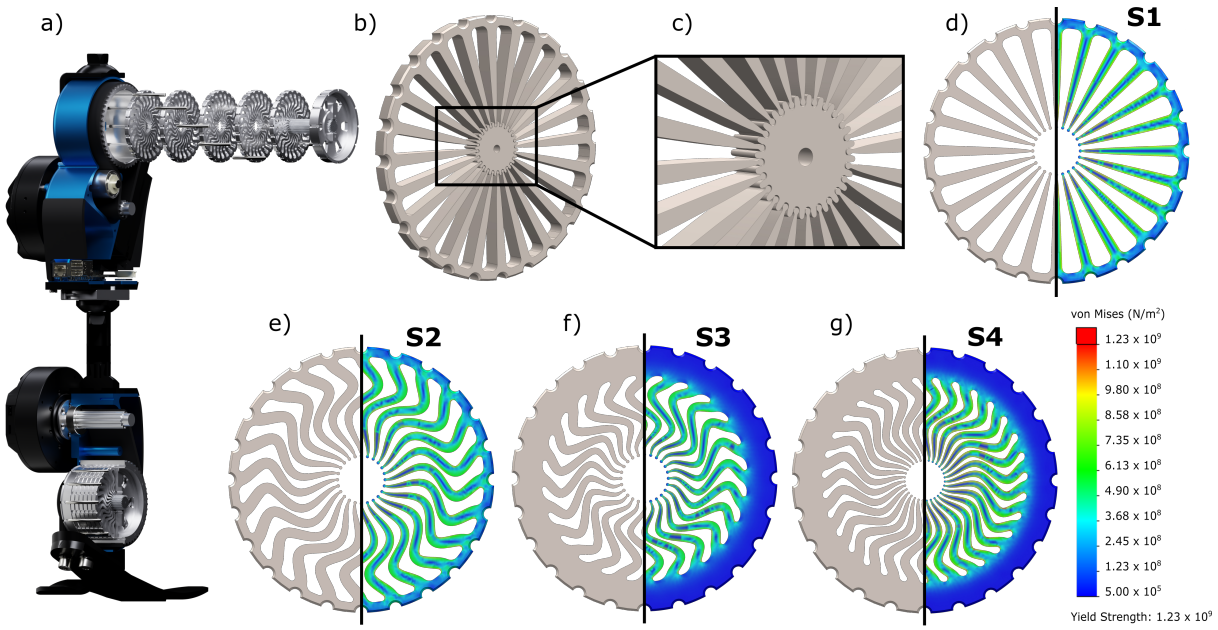

Figure 1: Rotary spring design based on bending cantilever flexures. a) the spring neatly mates with transmission components—in this case, a belt pulley in the knee and ankle of the Open-Source Leg v2. b) and c) the flexures mate with a gear-like camshaft, which loads the spring when rotated. d) the original design consists of straight flexures with a tapered profile. e) and f) later designs feature serpentine flexures in addition to the tapered profile. g) the number of flexures was also optimized for maximal energy storage in the spring.

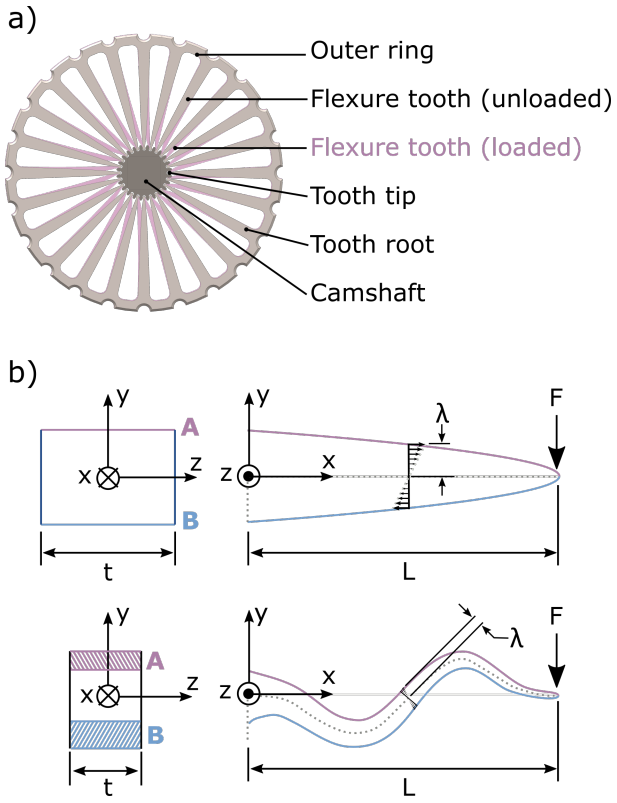

Figure 2: a) Spring with labeled parts. b) Schematic of beam coordinate frames and loading condition. Surfaces A and B are the bending surfaces.

Table 1: Design parameters and expected properties of the four springs. The mass of all four springs was calculated assuming a 2.5mm thick rim connecting all flexures on the outer edge.

Spring Design	**Spring Rate (Nm/rad)**	**Desired Deflection (rad)**	**Mass (g)**	**Root Radius (mm)**	**Contact Radius (mm)**	**Number of Flexures**	**Serpentine Factor**	**Density Factor**	**Cycles to Failur**
S1	150	0.220	57.3	31.0	6.0	24	1.00	0.40	100,00
S2	150	0.253	71.0	31.0	6.0	24	1.24	0.53	100,00
S3	150	0.211	51.8	26.0	6.0	24	1.32	0.53	100,00
S4	150	0.234	60.1	26.0	5.1	31	1.17	0.64	100,00

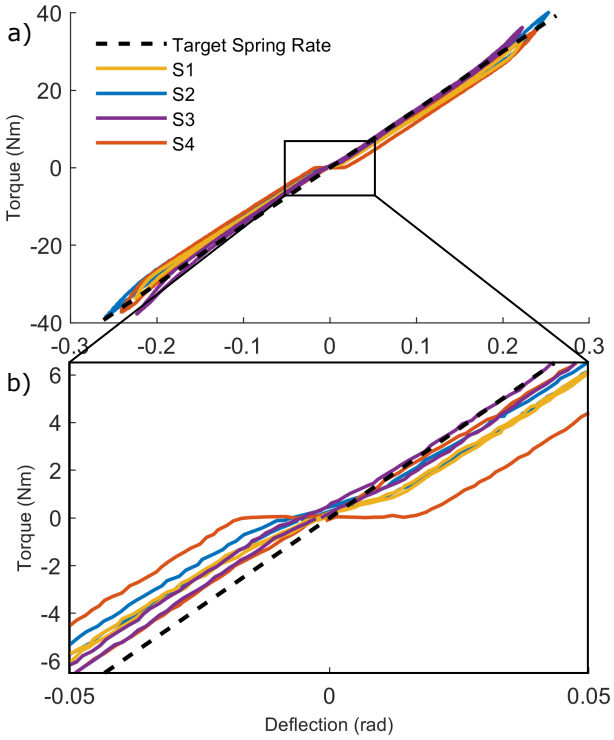

Figure 3: Measured spring rates of S1, S2, S3 and S4 over a) each spring's respective operable range and b) small deflections to show zero-torque crossover behavior. For reference, the target spring rate of 150 Nm/rad is also displayed.

Table 2: Spring rates of the four spring designs during loading and unloading in both positive and negative torque regimes. Average spring rate and percent error are also reported. The desired spring rate was 150 Nm/rad for all four designs.

Spring Design	**Spring Rate (Nm/rad)**				**Avg.**	**Error (%)**
	Loading		**Unloading**			
	Pos.	**Neg.**	**Pos.**	**Neg.**		
S1	149.2	150.9	139.4	139.9	144.9	3.43
S2	158.0	151.4	144.1	142.4	149.0	0.68
S3	156.6	171.5	152.1	159.9	160.0	6.68
S4	147.7	155.8	145.2	147.4	149.0	0.65

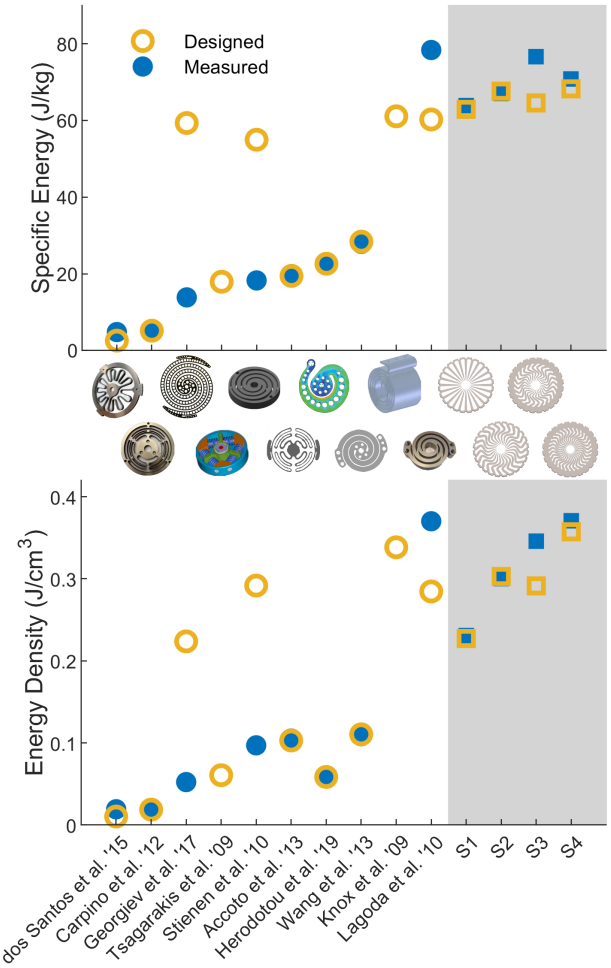

Figure 4: Performance of designed cantilever-beam rotary springs (gray region) by energy storage per unit mass and per unit volume in the context of the current literature. It should be noted that the mass and volume of the S3 spring was estimated for an outer rim thickness comparable to that of S1 and S2 rather than using the unnecessarily large rim than was dictated by our testing apparatus.

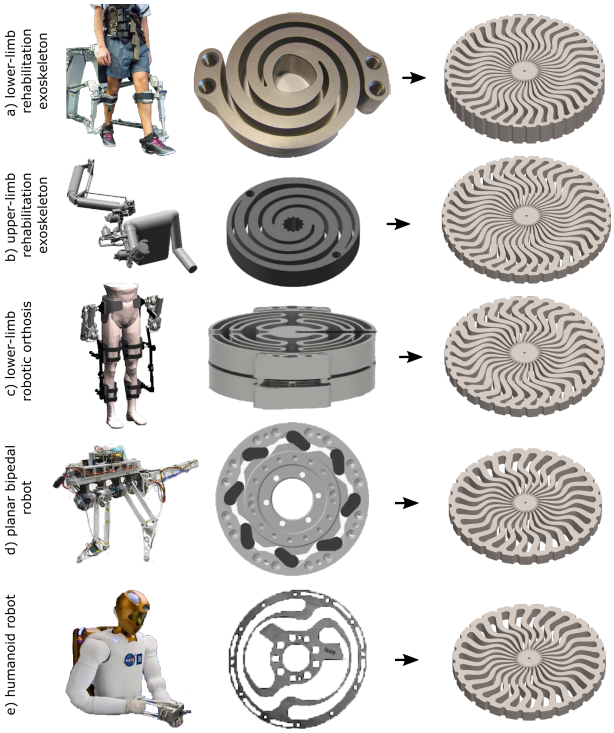

Figure 5: Effect of our novel spring architecture on wearable robots. In each row, the application is shown, followed by the original spring for that design and a spring designed by our open-source design tool with equivalent performance specifications. a) The LOPES II exoskeleton (*36*) uses series elastic actuators (*16*), b) the Limpact upper-limb rehabilitation exoskeleton uses a spring as part of a rotational hydroelastic actuator (*15*), c) a lower limb robotic orthosis (*5*) uses SEAs at the joints, d) a planar bipedal robot (*4*) uses rubber Neidhart springs, and e) the Robonaut 2 humanoid employs series elastic arms.

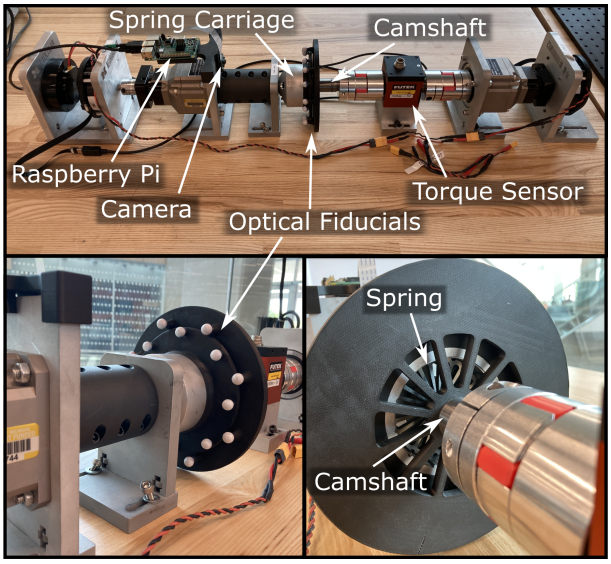

Figure 6: Testbed used to evaluate spring performance. By tracking optical fiducials with a camera, we were able to measure true deflection of the springs.

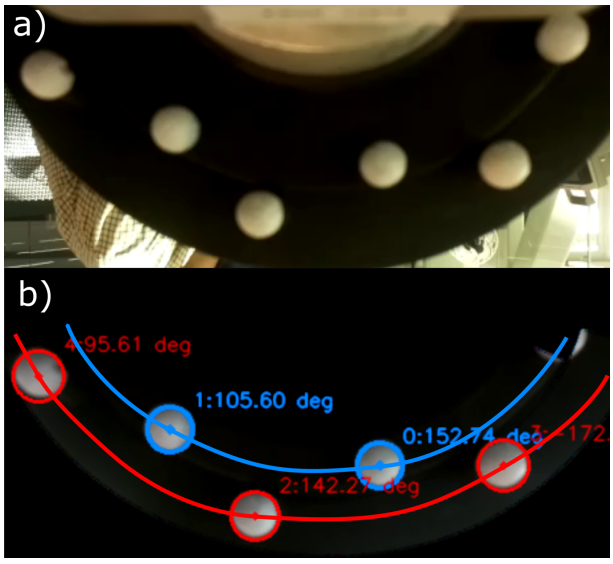

Figure 7: Testbed used to evaluate spring performance. By tracking optical fiducials with a camera, we were able to measure true deflection of the springs.

Table 3: Implementation of our spring design tool with various types of robots. For each of the listed applications, we used the open-source design tool to design a spring that matched the stiffness and deflection specifications of the robot, and compared the size and mass of our design to that of the original spring.

Application	**Ref.**	**Spring**	**Spring Rate (Nm/rad)**	**Defl. (rad)**	**Mass (g)**	**OD (mm)**	**Thick. (mm)**	**Improvement (%)**	
								Mass	**Volume**
Lower-limb rehab exo	(*16*)	Original	353	0.283	235	65	15	12	23
		New Design			206	70	10		
Upper-limb rehab exo	(*15*)	Original	150	0.332	150	60	10	21	24
		New Design			119	74	5		
Lower-limb robotic orthosis	(*5*)	Original	250	0.240	370	90	23.5	71	87
		New Design			109	70	5		
Planar Bipedal Robot	(*4*)	Original	150	0.262	150	67	25	43	80
		New Design			85.3	67	5		
Humanoid Robot	(*33*)	Original							
		New Design							