ON OCTOPUS ARM AND MULTIGRID TECHNIQUES FOR NONLINEAR CONSTRAINED MINIMIZATION

DORIT RON*, TAMAR FLASH†, AND ACHI BRANDT‡

Abstract. Our objective is to calculate a curve (configuration) in two dimensions that solves an optimization problem of minimizing a generalized squared curvature functional subject to global constrained. We allow the given rigidity coefficient to vary along the curve. The global constraints include the length of the curve and its initial and ending positions and directions. We use a multigrid approach to efficiently solve the resulting discretized system of non-linear equations. This approach is in particular convenient for introducing the desired global constraints and solving the entire problem in linear time complexity. Since our work is motivated by the idea of building a soft octopus-like robotic arm, we then produce a dynamic sequence of such configurations depending on time to resemble a complete movement of the octopus arm, for instance in a reaching movement. The obtained sequence involves bend propagation as well as elongation of the arm as is typical to the octopus. We suggest the use of our results as input to the robotic arm in order to enable its basic movements. The numerical approach itself can be generalized to much more complex problems in higher dimensions.

Key words. constrained optimization, multigrid, minimal squared curvature, octopus arm extension, soft robotic arm

1. Introduction. For many years scientists and engineers have investigated continuum robots [21]. Compared with traditional rigid-body structures, continuum robots with their large number of degrees of freedom are capable of flexible and highly dexterous movements. An example in nature of a continuum structure are octopus arms [23]. The octopus arm in particular has no internal or external skeleton and its muscular structure is considered to be a "muscular hydrostat" [18]. Its arms are fully flexible, can bend in any direction and grasp objects. Therefore a continuum arm design inspired by octopus (Octopus vulgaris) would be suitable for searching in unsafe and narrow spaces [26] [25], and surgical applications [22].

The trajectory generation problem for mobile robots in general and for the robotic arm inspired by octopus in particular, is of purely geometric nature and can be defined as providing a set of time dependent configurations that are "smooth" and meet certain boundary conditions. The notion of "smoothness" is an ambiguous one. Authors have used different types of smoothness criteria in order to derive trajectories. Kanayama et al. [17] used the square of curvature and the square of the derivative of curvature as cost functions. The configurations that minimize these criteria are clothoids and cubic spirals. In fact, variations of the problem of minimizing the squared curvature have a long history and various applications, see for example [20] and [19]. Horn [15] addressed the problem of determining a smooth planar configuration that started at $(0, 0)$ in the vertical direction arrived at $(1, 0)$ vertically and minimized the integral of the squared curvature. He derived a differential equation of the optimal configuration along with its properties and concluded that the optimal

*Faculty of Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot 76100, Israel dorit.ron@weizmann.ac.il

†Faculty of Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot 76100, Israel tamar.flash@weizmann.ac.il

‡Faculty of Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot 76100, Israel achi.brandt@weizmann.ac.il

one is not a semi circle. Later, Kallay [16] solved the problem where the length of the solution is also constrained. Bruckstein et al. [11] introduced a scale-invariant measure to avoid solutions which are smoother just by being longer, and showed that due to this measure circular arcs are optimal in large number of situations. They proposed a numerical procedure for computing piecewise linear approximations of the optimal configurations as a solution of discrete two-point boundary value problems.

In this paper we present a fast multilevel solver for the *generalized* problem of minimizing the squared curvature of a curve, given its length and starting and ending positions and directions. In addition, we allow the curve to have an arbitrary variable *rigidity coefficients*, which can control the curvature function produced in the resulting configuration, with smaller curvature wherever the coefficient is larger. This enables solutions with *variable* curvature, i.e., configurations which are in agreement with the observed typical octopus arm movement (composed of a sequence of time dependent configurations). We show that the bending movement of the robotic arm prototype in [29] actually minimizes the squared curvature with *uniform* rigidity coefficients, while the octopus arm extension movement which usually includes bend propagation can be described only when *variable* rigidity coefficients are employed. We achieved good resemblance to the octopus arm movement.

The multilevel solver we introduce here is a linear time algorithm (in the number of discrete unknowns). In fact it can produce a sequence of solutions much faster than and in a resolution which is beyond one would actually need for the robotic arm. However, we regard this solver as a stepping stone for treating more challenging problems, for which the introduced technology is essential.

The paper is organized as follows. In Section 2 we introduce the optimization problem to be solved. In Section 3 we present the basic multilevel techniques used for the fast solver. In Section 4 we show the resemblance of our results to the bending movement of the robotic arm and to the bend propagation of the octopus arm. Finally, in Section 5 we outline various problem directions, including control and other inverse problems, for which extensions of the multigrid solver developed here can apply.

2. The optimization problem. Our objective is to minimize the two dimensional squared curvature of a curve with a given length L subject to boundary conditions stating its starting and ending positions and directions. We will consider the normalized problem where we choose the starting point to be $(x_{\mathbf{start}}, y_{\mathbf{start}}) = (0,0)$ ending at $(x_{\mathbf{end}}, y_{\mathbf{end}}) = (1,0)$:

$$\begin{array}{ll} \textbf{minimize} & E(a,u) = \frac{1}{2L}\int_0^1 a(s)(u'(s))^2 ds, \\ \textbf{given} & u(0) = u_{\mathbf{start}}\ ,\ u(1) = u_{\mathbf{end}} \\ \textbf{subject to} & L\int_0^1 \cos u(s)\ ds = 1\ ,\ L\int_0^1 \sin u(s)\ ds = 0, \end{array} \tag{2.1}$$

where s is the arclength along the curve divided by L, $u(s)$ is the tangent of the curve at s, $(u'(s))^2$ is thus the squared curvature and $a(s)$ is the rigidity coefficient. For given larger values of $a(s)$ the obtained solution is expected to be less curved at s.

We discretize L to have $n-1$ segments of (usually) equal length h (see on the bottom of Figure 3.1 a mesh with 12 segments) and use two Lagrange multipliers λ_1 and λ_2 to obtain the following discrete Lagrangian

$$\begin{array}{rl} \mathcal{L}(h,a,u) & = \frac{1}{2Lh}\sum_{j=1}^{n-1} a_{j+1/2}(u_{j+1}-u_j)^2 + \\ & \lambda_1[Lh(\frac{1}{2}\cos u_1 + \sum_{j=2}^{n-1}\cos u_j + \frac{1}{2}\cos u_n) - 1] + \\ & \lambda_2 Lh(\frac{1}{2}\sin u_1 + \sum_{j=2}^{n-1}\sin u_j + \frac{1}{2}\sin u_n), \end{array} \tag{2.2}$$

where $u_j = u(s_j), s_j = (j-1)h$ and $a_{j+1/2}$ is the rigidity coefficient between u_j and u_{j+1}.

The resulting system of non-linear equations obtained from $\partial\mathcal{L}(h,a,u)/\partial u_i = 0$ and from $\partial\mathcal{L}(h,a,u)/\partial\lambda_k = 0$, for $k = 1,2$ is

$$\begin{array}{ll} \textbf{for } i = 2,...,n-1 & N^1(h,a,u,i) \equiv \\ & (-\overline{a}_{i+1/2}u_{i+1} + (\overline{a}_{i+1/2} + \overline{a}_{i-1/2})u_i - \overline{a}_{i-1/2}u_{i-1})/h) \\ & \qquad - L^2\lambda_1 \sin u_i + L^2\lambda_2 \cos u_i = b_i^1 \\ \textbf{given} & u(0) = u_1 = u_{\textbf{start}}\ ,\ u(1) = u_n = u_{\textbf{end}} \\ \textbf{subject to} & N^2(h,u) \equiv \\ & Lh(\frac{1}{2}\cos u_1 + \sum_{j=2}^{n-1}\cos u_j + \frac{1}{2}\cos u_n) = b^2, \\ & N^3(h,u) \equiv \\ & Lh(\frac{1}{2}\sin u_1 + \sum_{j=2}^{n-1}\sin u_j + \frac{1}{2}\sin u_n) = b^3, \end{array} \tag{2.3}$$

where $\overline{a}_{i+1/2} = a_{i+1/2}/h$, $b_i^1 = b^3 = 0$ and $b^2 = 1$.

An alternative way to derive the above is by variation of the differential minimization problem. Assume the minimum is obtained at some $u(s)$ and look at a perturbation $\varepsilon(s)$ to it such that $\varepsilon(0) = \varepsilon(1) = 0$. We use Lagrange multipliers and consider the following difference

$$\begin{array}{ll} E(a,u+\varepsilon) - E(a,u) & = \frac{1}{L}\int_0^1 a(s)u'(s)\varepsilon'(s)ds - L\lambda_1\int_0^1 \varepsilon(s)\sin u(s)ds \\ & \qquad + L\lambda_2\int_0^1 \varepsilon(s)\cos u(s)ds + O(\varepsilon(s)^2) \\ & = -\frac{1}{L}\int_0^1 [a(s)u'(s)]'\varepsilon(s)ds - L\lambda_1\int_0^1 \varepsilon(s)\sin u(s)ds \\ & \qquad + L\lambda_2\int_0^1 \varepsilon(s)\cos u(s)ds + O(\varepsilon(s)^2), \end{array} \tag{2.4}$$

where we have used integration by parts for the first term of the right hand side. Since this difference has to be non-negative for any $\varepsilon(s)$, we obtain the following differential equation

$$-[a(s)u'(s)]' - L^2\lambda_1\sin u + L^2\lambda_2\cos u = 0, \tag{2.5}$$

subject to the same conditions of (2.1). (2.3) is obtained by central discretization of (2.5).

3. The multilevel solver. We use multigrid techniques [10], [2] and [24] to efficiently solve (2.3), the resulting discretized system. We start by describing a point-by-point minimization of (2.1), then choose a hierarchy of grids and define the coarse-to-fine interpolation rule which is based on minimization consideration. The calculation of the coarse grid equations and the fine-to-coarse transfer of residuals follow. We use the Full-Approximation-Scheme (FAS) due to the non-linearity of the system of equations, imposed by the non-linear global constraints, and employ the Full-Multigrid (FMG) algorithm. In addition, we use continuation of the boundary conditions to enable a stable solution for more curved configurations.

3.1. Relaxation. A point-by-point minimization of (2.1) is equivalent to solving in turn the i'th equation of system (2.3) for u_i. However, since each equation is *nonlinear* function of u_i, the relaxation of each u_i should not waste work on solving the relevant equation *exactly*. Rather, only an inexpensive substantial step toward such a solution should be made. In particular, if $\widetilde{u}_i$ is the current value of u_i, then its value after relaxing the i'th equation is

$$\widetilde{u}_i \leftarrow [\overline{a}_{i+1/2}\widetilde{u}_{i+1} + \overline{a}_{i-1/2}\widetilde{u}_{i-1} + (b_i^1 + L^2\lambda_1\sin\widetilde{u}_i - L^2\lambda_2\cos\widetilde{u}_i)h]/(\overline{a}_{i+1/2} + \overline{a}_{i-1/2}). \tag{3.1}$$

Note that the trigonometric terms of $\widetilde{u}_i$ are "non-principal terms" as they are multiplied by h and thus their contribution to the change in the residual can be neglected.

As is well known [9], following a small number of relaxation sweeps, the remaining error $e = u - \widetilde{u}$ is smooth and therefore can be approximated by a "coarser" (or "diluted") system, i.e., a system with less variables.

3.2. Hierarchy of grids. In the classical case of "geometric multigrid", where the fine-level set of unknows u is defined on a well-structured grid, the coarse set U is naturally defined in terms of coarsening that grid, for example by omitting from it every other point, see the two bottom grids in Figure 3.1. All finer grids are uniform such that the mesh of a finer grid is twice as smaller as the coarser one. To formulate the relation between two adjacent levels of our multigrid solver, we use the convention of describing the fine level by lower case letters such as h, u_i and $a_{i+1/2}$ and the coarse grid by capitals: H, U_I and $A_{I+1/2}$. The relation between the fine and the coarse mesh sizes is then $H = 2h$.

The needed *finest* mesh size depends on the prescribed boundary angles. If, for instance, the absolute value of the initial angle exceeds $\Pi/2$, the curve has to turn around to reach the other end, as can be seen at both ends of the black curve in Figure 4.4. In particular, if L is close to 1, the angles at the "bent" end of the curve has to change quickly within a small arc length. This would imply the use of smaller h towards that end. The sharper the bent, the smaller the needed h [1]. In this work we do not intend to solve all possible combinations of L and boundary angles, but we rather assume that the bent cannot be too sharp, which is reasonable for both the octopus and the robotic arm. Still, to enable more curvature at the beginning and end of our solution (as is usually observed at the octopus arm) we choose an *uneven* coarsest level as shown in Figure 3.1, where the mesh size is smaller near the end points than the one in the middle. For this *nonuniform* mesh (2.3) takes the form

$$
\begin{aligned}
&\textbf{for } i = 2,3 \quad && N^1(h,a,u,i) \equiv \\
&&& (-\overline{a}_{i+1/2}u_{i+1} + (\overline{a}_{i+1/2} + \overline{a}_{i-1/2})u_i - \overline{a}_{i-1/2}u_{i-1})/(\tfrac{h_{i+1/2}+h_{i-1/2}}{2}) \\
&&& \quad - L^2\lambda_1 \sin u_i + L^2\lambda_2 \cos u_i = b_i^1 \\
&\textbf{given} && u(0) = u_1 = u_{\textbf{start}} \ , \ u(1) = u_4 = u_{\textbf{end}} \\
&\textbf{subject to} && N^2(h,u) \equiv \\
&&& L(\cos u_1 \tfrac{h_{3/2}}{2} + \cos u_2 \tfrac{h_{3/2}+h_{5/2}}{2} + \cos u_3 \tfrac{h_{5/2}+h_{7/2}}{2} + \cos u_4 \tfrac{h_{7/2}}{2}) = b^2, \\
&&& N^3(h,u) \equiv \\
&&& L(\sin u_1 \tfrac{h_{3/2}}{2} + \sin u_2 \tfrac{h_{3/2}+h_{5/2}}{2} + \sin u_3 \tfrac{h_{5/2}+h_{7/2}}{2} + \sin u_4 \tfrac{h_{7/2}}{2}) = b^3,
\end{aligned}
\tag{2.3'}
$$

where $\overline{a}_{i+1/2} = a_{i+1/2}/h_{i+1/2}$ and $h_{i+1/2}$ is the i'th mesh size, $i = 1,2,3$.

In an *adaptive* approach instead of choosing every other fine point to serve as the coarse points, we may choose all (or most) fine points wherever a relatively large difference between neighboring unknowns is detected. Similarly, less (than the usual half) grid points are needed where the solution is smooth. For full efficiency of the solver, all grids should actually be uneven, chosen according to the detected curvature at each site. Then the choice of the coarse grid can be completely automated by using the so called bootstrap algebraic multigrid (BAMG) [5]. But we do not attempt this generality here.

3.3. Minimization based interpolation. The derivation of our specific interpolation from a pair of coarse grid points to possibly *many* inner fine points, as is for

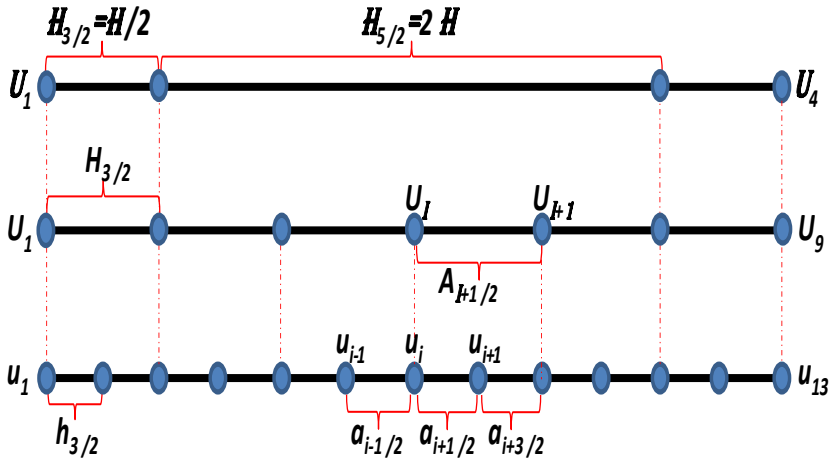

FIG. 3.1. *An example of three grids where the coarsest of them is non uniform with grid points closer to the end points. Still the coarsening ratio is preserved:* $H = 2h$ *and* $\boldsymbol{H} = 2H$.

example needed (in the middle of the grid) between the two coarse levels in Figure 3.1, is given in the Appendix.

For the common coarsening ratio 2, where the coarse grid points are chosen to be at every other fine point (the two finest levels of Figure 3.1), the interpolation to u_{i+1}, which is between two coarse grid points, is

$$u_{i+1} \leftarrow u_{i+1} + (\uparrow_H^h V)_{i+1} = u_{i+1} + \frac{V_I a_{i+1/2} + V_{I+1} a_{i+3/2}}{a_{i+1/2} + a_{i+3/2}}, \tag{3.2}$$

where V is the coarse correction to the fine grid function u. The interpolation to u_i is simply $u_i \leftarrow u_i + V_I$. We use equation (3.2) for the interpolation except from the coarsest level, where equation (A.5) is needed.

3.4. Coarsening. The derivation of the coarse grid equations given an explicit coarse-to-fine interpolation operator is simply the equations for minimizing $E(u+ \uparrow_H^h V)$. Thus we calculate the coarse couplings $A_{I+1/2}$ as well as the right hand side of the coarse equations. Consider the generalized optimization problem

$$\textbf{minimize} \quad E(a,b,u) = \frac{1}{2}\int a(s)(u'(s))^2 ds - \int b(s)u(s)ds, \tag{3.3}$$

and its discretization

$$\textbf{minimize} \quad E^h(a,b,u) = \frac{1}{2}\sum_{j=1}^{n-1} \frac{a_{j+1/2}(u_{j+1}-u_j)^2}{h_{j+1/2}} - \sum_{j=1}^{n-1} b_j u_j \frac{h_{j+1/2}+h_{j-1/2}}{2}. \tag{3.4}$$

On the finest grid $b_j = 0$ for $j = 1, ..., n-1$. The equivalent j'th equation is

$$-\overline{a}_{j+1/2}u_{j+1} + (\overline{a}_{j+1/2} + \overline{a}_{j-1/2})u_j - \overline{a}_{j-1}u_{j-1/2} = b_j \frac{h_{j+1/2}+h_{j-1/2}}{2}, \tag{3.5}$$

so the j'th *residual* can be defined as

$$\mathbf{res}_j \;=\; b_j \frac{h_{j+1/2}+h_{j-1/2}}{2} + \overline{a}_{j+1/2}u_{j+1} - (\overline{a}_{j+1/2}+\overline{a}_{j-1/2})u_j + \overline{a}_{j-1/2}u_{j-1}. \quad (3.6)$$

Given the set of fine and coarse mesh sizes, $h_{i+1/2}$ and $H_{I+1/2}$, see Figure 3.2, we want to calculate the equation at the coarse grid point I. For a given current fine approximation $\widetilde{u}$, using (A.5) and according to Figure 3.2, we get

$$\begin{aligned} E^h(a,b,\widetilde{u}+\uparrow_H^h V) = \;& \tfrac{1}{2}\{\textstyle\sum_{j=l}^{i-1}\overline{a}_{j+1/2}[\widetilde{u}_{j+1}+(\sum_{k=j+1}^{i-1}\alpha_k V_{I-1}+\sum_{k=l}^{j}\alpha_k V_I)/a_{li} - \\ & \widetilde{u}_j \;-(\textstyle\sum_{k=j}^{i-1}\alpha_k V_{I-1}+\sum_{k=l}^{j-1}\alpha_k V_I)/a_{li}]^2 + \\ & \textstyle\sum_{j=i}^{n-1}\overline{a}_{j+1/2}[\widetilde{u}_{j+1}+(\sum_{k=j+1}^{n-1}\alpha_k V_I+\sum_{k=i}^{j}\alpha_k V_{I+1})/a_{in} - \\ & \widetilde{u}_j \;-(\textstyle\sum_{k=j}^{n-1}\alpha_k V_I+\sum_{k=i}^{j-1}\alpha_k V_{I+1})/a_{in}]^2\} - \\ & \textstyle\sum_{j=l}^{i-1} b_j h_{j+1/2}\,[\widetilde{u}_j+(\sum_{k=j}^{i-1}\alpha_k V_{I-1}+\sum_{k=l}^{j-1}\alpha_k V_I)/a_{li}] - \\ & \textstyle\sum_{j=i}^{n-1} b_j h_{j+1/2}[\widetilde{u}_j+(\sum_{k=j}^{n-1}\alpha_k V_I+\sum_{k=i}^{j-1}\alpha_k V_{I+1})/a_{in}], \end{aligned} \quad (3.7)$$

where, $a_{li}=\sum_{k=l}^{i-1}\alpha_k$, $a_{in}=\sum_{k=i}^{n-1}\alpha_k$ and $\alpha_k = 1/\overline{a}_{k+1/2}$. Analogously to the fine grid equation (3.5), the resulting coarse grid equation at the coarse site I is

$$-\overline{A}_{I+1/2}V_{I+1}+(\overline{A}_{I+1/2}+\overline{A}_{I-1/2})V_I-\overline{A}_{I-1/2}V_{I-1} = B_I\frac{H_{I+1/2}+H_{I-1/2}}{2}, \quad (3.8)$$

where

$$\overline{A}_{I+1/2} = (\sum_{k=i}^{n-1}\alpha_k)^{-1}, \quad (3.9')$$

$$B_I = 2\sum_{j=l+1}^{n-1}[(\uparrow_h^H)_j^I \mathbf{res}_j]/(H_{I+1/2}+H_{I-1/2}), \quad (3.10')$$

$H_{I-1/2}=\sum_{k=l}^{i-1}h_{k+1/2}$, $H_{I+1/2}=\sum_{k=i}^{n-1}h_{k+1/2}$ and the residual weighting coefficients are

$$(\uparrow_h^H)_j^I \;=\; \begin{cases} \sum_{k=l}^{j-1}\alpha_k & l<j<i \\ 1 & j=i \\ \sum_{k=j}^{n-1}\alpha_k & i<j<n. \end{cases} \quad (3.11')$$

Again, we only use this generalized form to deduce the equations of the coarsest level. For the simpler uniform grids case, we get

$$\overline{A}_{I+1/2} = (1/\overline{a}_{i-1/2}+1/\overline{a}_{i+1/2})^{-1}, \quad (3.9)$$

$$B_I = \frac{1}{H}\sum_{j=i-1}^{i+1}[(\uparrow_h^H)_j^I \mathbf{res}_j] \quad (3.10)$$

and

$$(\uparrow_h^H)_j^I \;=\; \begin{cases} 1/\overline{a}_{i-1/2} & j=i-1 \\ 1 & j=i \\ 1/\overline{a}_{i+1/2} & j=i+1. \end{cases} \quad (3.11)$$

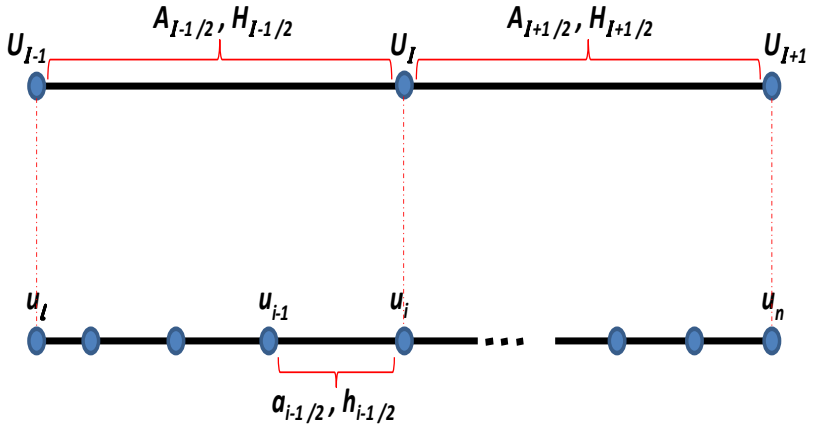

FIG. 3.2. *An example of corresponding sections of two grids. The coefficient $A_{I+1/2}$ is calculated from all $a_{i+1/2}$'s between i and n and the transfer of residuals from the fine level to the coarse one at site I involves the residuals at all fine points between sites $l+1$ and $n-1$ as is calculated in (3.11').*

3.5. Full approximation scheme (FAS). Since the system of equations (2.3) that we want to solve is nonlinear, due to its global constraints, we actually need to use the Full Approximation Scheme (FAS), introduced in [2], which is called so because it gives the coarse equations in terms of the full solution $U = \widetilde{U} + V$, not in terms of the correction V alone [9], where $\widetilde{U}$ is the current approximate fine solution $\widetilde{u}$ transferred to the coarse level. Namely,

$$\widetilde{U}_I = \sum_{j=l+1}^{n-1} (\uparrow_h^H)_j^I \widetilde{u}_j, \tag{3.12}$$

where l (n) is the fine index at the grid point $I-1$ ($I+1$), $(\uparrow_h^H)_j^I$ is given by (3.11') (or by (3.11)) and the boundary values are transferred directly.

Using for the linear part of the equations the operator suggested by equation (3.8), and applying to it the standard FAS procedure, yields a coarse problem similar to the fine one (2.3):

$$\begin{array}{ll} \textbf{for } I = 2, ..., N-1 & N^1(H, A, U, I) = B_I^1 \\ \textbf{given} & U(0) = U_1 = u_{\textbf{start}}\ ,\ U(1) = U_N = u_{\textbf{end}} \\ \textbf{subject to} & N^2(H, U) = B^2, N^3(H, U) = B^3, \end{array} \tag{3.13}$$

where $\overline{A}_{I+1/2} = A_{I+1/2}/H_{I+1/2}$ is given by (3.9') (or by (3.9)) and the coarse right hand sides are

$$B_I^1 = \sum_{j=l+1}^{n-1} (\uparrow_h^H)_j^I [b_i^1 - N^1(h, a, \widetilde{u}, j)] + N^1(H, A, \widetilde{U}, I). \tag{3.14}$$

and

$$B^q = [b^q - N^q(h, \widetilde{u})] + N^q(H, \widetilde{U}), \qquad (q = 2, 3). \tag{3.15}$$

It is important to remember that it is the *error* $e = u - U$ which is smoothed by relaxation and thus approximated by the correction $V = U - \widetilde{U}$, so after obtaining an approximate solution U to the coarse equation (3.13), the coarse-grid correction to the fine level is

$$\widetilde{u} \leftarrow \widetilde{u} + \uparrow_H^h (U - \widetilde{U}) \ , \tag{3.16}$$

where $\uparrow_H^h$ is given by (A.5) (or by (3.2)). It is also important to use exactly the same $\widetilde{U}$ in (3.16) as in (3.14), (3.15) and as the first approximation in the process of solving (3.13), so that the solver is *stationary*, meaning that if the exact solution has already been obtained at the fine level ($N^1(h, a, \widetilde{u}, i) = b_i^1$) and hence (3.13) is immediately at its solution, then (3.16) does not change $\widetilde{u}$.

3.6. Multigrid cycle. Having constructed the coarse-level equations, they are then (approximately) solved by a similar procedure: a small number of relaxation sweeps followed by approximating the remaining error with a still coarser system. The coarsest-level equations are finally solved by some direct method. This recursively defines the multilevel cycle, which, for a work comparable to that of just few relaxation sweeps over the finest level (the given system), would usually reduce the error to a small fraction (far less than .5, typically) of its pre-cycle size. Such a cycle is called a *V-cycle* due to its V shape as can be seen in the right of Figure 3.3.

In our solver the coarsest level includes five interior points (unknowns), the middle grid in Figure 3.1. To solve (2.3) we use Newton-Raphson iterations. To obtain a good first guess, we actually start with an even coarser grid which includes only two interior points, the top grid in Figure 3.1. The corresponding two unknowns, u_2 and u_3, can be calculated from the two (non linear) last conditions in (2.3'). Then the two Lagrange multipliers are extracted from the remaining two (linear) equations. and are kept unchanged on all fine levels and may be corrected only at the coarsest level. Thus the two global constraints are actually being solved only on the coarsest level, while on finer grids their *residuals* are being calculated and transferred to coarser and coarser grids until the coarsest one is reached.

Full Multigrid (FMG) algorithm. The multigrid cycle described above can be applied to any first approximation given on the finest grid. In a **f**ull **m**ulti**g**rid (FMG) algorithm, the first approximation is obtained by interpolation from a solution on the next coarser grid, which has previously been calculated by a similar FMG algorithm. A typical FMG algorithm, with one V-cycle per refinement, is shown in Figure 3.3. This is the algorithm we use here. In particular, the FMG algorithm begins by solving the coarsest level explicitly.

3.7. Continuation. It is easy to solve the problem for certain initial end angles ($u_{\mathbf{start}}$ and $u_{\mathbf{end}}$) and much harder for others. We have thus used a *continuation* approach to solve more difficult cases gradually, starting with easy-to-solve problems. For example, consider a circular arc of length $L > 1$ which extends over a chord of length 1, both resting on a central angle $2u$, so that $L = 2ru$; see Figure 3.4. The radius of the circle is $r = 0.5/\sin u$, so u satisfies $u/\sin u = L$ (or approximately $u \sim \sqrt{6(L-1)/L}$ when small). We can thus use such an arc as our initial solution, with $u_{\mathbf{start}} = -u_{\mathbf{end}} = u$. Having solved this problem we gradually change the boundary angles to actually obtain a solution to any given problem. This is mostly done on the coarsest grid (the one with five interior unknowns) by using a Newton-Raphson's iteration successively for a sequence of problems, the (approximate) solution of each

one serves as the initial guess for the next one, while the boundary angles are gradually changed until the desired ones are met. For large $u_{\mathbf{start}}$ and/or $-u_{\mathbf{end}}$ one has to do this continuation on a finer level, using our multigrid cycle in each iteration.

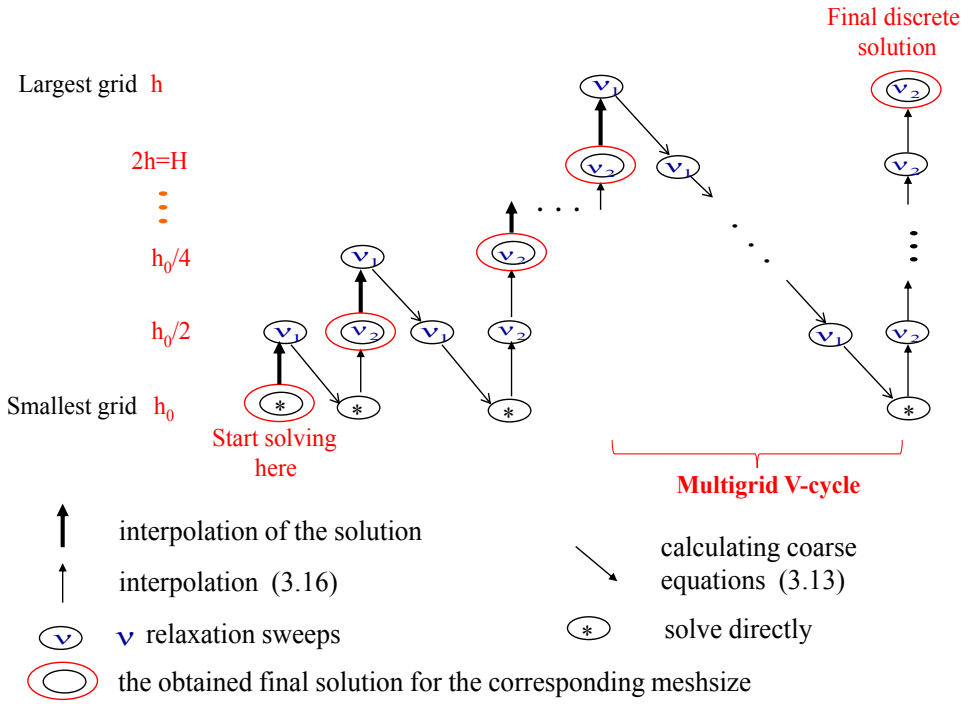

FIG. 3.3. *An FMG-cycle: starting from a direct solution of the coarsest level (on the left), interpolate to a finer level, apply a V-cycle there and switch to a still finer level, etc. A V-cycle $V(\nu_1,\nu_2)$ (on the right): starting from the finest level, perform ν_1 relaxation sweeps, then transfer to the next coarser level and so on until the coarsest level. Solve the coarsest level exactly, then interpolate to a finer grid followed by ν_2 relaxation. This is repeated until the finest level is reached.*

4. Results.

4.1. The efficiency of the cycle. The efficiency of a multigrid cycle is often measured by calculating residuals. We calculate here the residual of the system of equations in (2.3), i.e.,

$$\mathbf{Res}^1 \equiv \sqrt{\sum_i^n (b_i^1 - N^1(h,a,u,i))^2} \quad , \tag{4.1}$$

and the residuals of the two constraints:

$$\mathbf{Res}^2 \equiv |b^2 - N^2(h,u)| \quad , \quad \mathbf{Res}^3 \equiv |b^3 - N^3(h,u)|. \tag{4.2}$$

We use the FMG (with one V-cycle per each level) to obtain the solution on the finest level followed by additional V-cycles only on that level. As can be seen in Table 4.1 the residuals are decreased by a factor of at least 10 per cycle which is the desired textbook efficiency of multigrid.

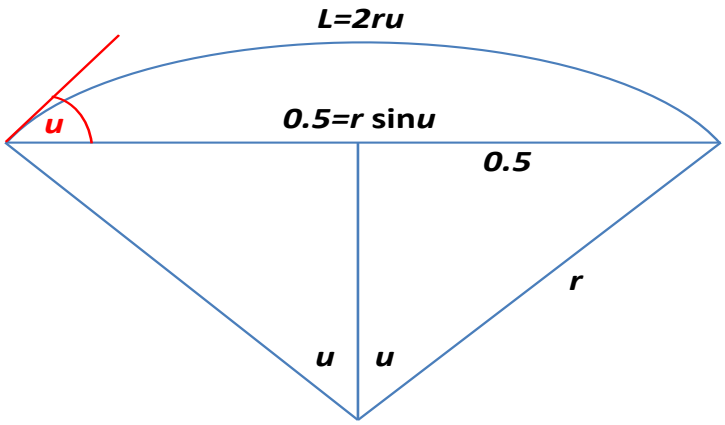

FIG. 3.4. *An easy-to-solve problem: a circular arc of length $L > 1$ which extends over a chord of length 1, both resting on a central angle $2u$. The angle between the chord and the tangent (the red line), which is also u, can be obtained by solving $u/\sin u = L$.*

Cycle number	$\mathbf{Res}^1$	$\mathbf{Res}^2$	$\mathbf{Res}^3$
1	0.491258	0.000833	0.002888
2	0.026529	-0.000043	0.000032
3	0.001547	-0.000006	0.000004
4	0.000093	0.000000	-0.000001

TABLE 4.1

The obtained residuals for the system of equations and for the two constraints are shown for four V-cycle $V(2,1)$.

4.2. The resemblance to a robotic arm. Soft robotic arm inspired by the octopus has been designed and tested for certain movements [29]. A two dimensional reconstruction of a robotic arm prototype *bending* movement (obtained at its early stage of development), using an appropriate algorithm [28], was carried out and used to detect the robotic arm backbone from imaged video frames as is shown in Figure 4.1. We were able to imitate the initial and final configurations in this figure, see Figure 4.2, and in fact the entire movement as shown in Figure 4.3. We may conclude that every configuration within this robotic arm movement (at least roughly) actually minimizes our mathematical model (2.3), i.e., minimizing the squared curvature given the length of the robotic arm and the positions and angles of its end points for *uniform constant* rigidity coefficients.

4.3. The resemblance to a single octopus arm configuration. Octopus arm movements were video recorded while an octopus was kept in a large glass water tank. For details on animals, experimental setup and recording sessions see [13] and [27]. Zelman et al. [28] reconstructed the octopus arm movement and produced a sequence of three dimensional curves varying in time. Each time instance is given as a set of 100 3D points. Since it has been observed that most of the extension movement of the octopus's arm is basically planar [13], we projected the three dimensional data into two dimensions by using a principal components reduction. Indeed, it turns out that about 95% (and sometimes even more) of the curve is planar. In addition, we have

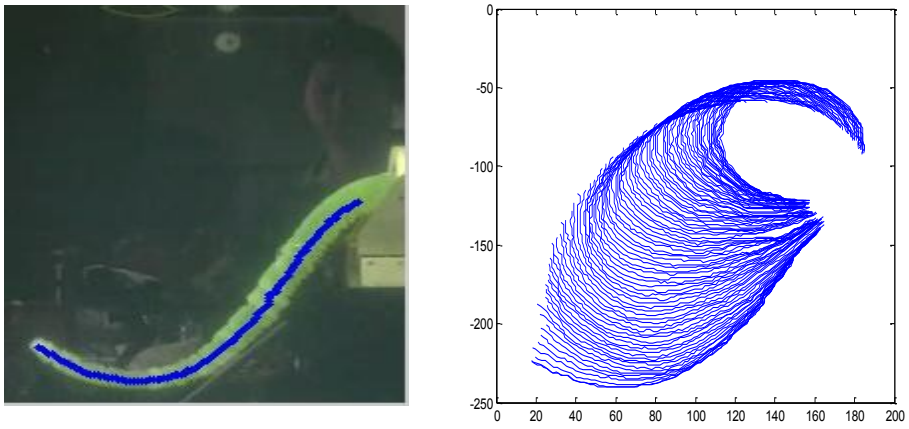

FIG. 4.1. *The robotic arm is performing a bend movement starting from the initial position shown on the left. The right picture in obtained by reconstructing the backbone of the arm from many imaged video frames taken while the bent was performed.*

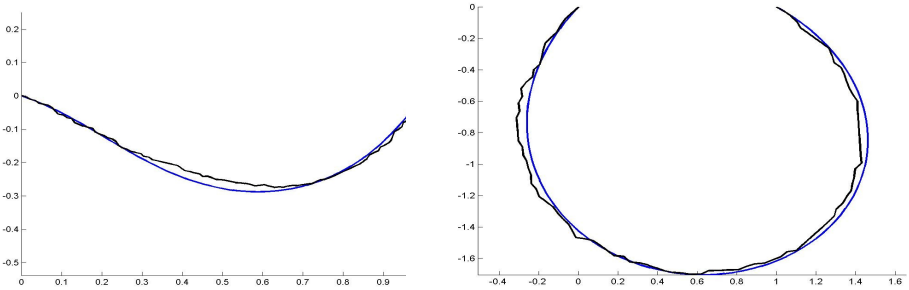

FIG. 4.2. *Resemblance of the initial and the final configurations of the robotic arm movement and the solution of minimizing the squared curvature with uniform rigidity coefficients.*

normalized the starting point to $(0,0)$ and the ending point to $(1,0)$. The length and the angles boundary conditions are easily measured. Solving the obtained problem in (2.3) on a grid with 24 meshes and constant rigidity ($a_i \equiv 1$ for $i = 1, ..., 24$) results with an evenly curved solution shown on the left of Figure4.4. If, however, we want to obtain a better agreement to the octopus arm, we need to introduce *variable* rigidity coefficients to enable the variable rigidity exhibited by the octopus arm; see at the right of Figure 4.4 for the choice of $a_1 : a_5 = 0.5$, $a_6 : a_8 = 0.1$, $a_{21} : a_{22} = 0.1$, $a_{23} : a_{24} = 0.2$.

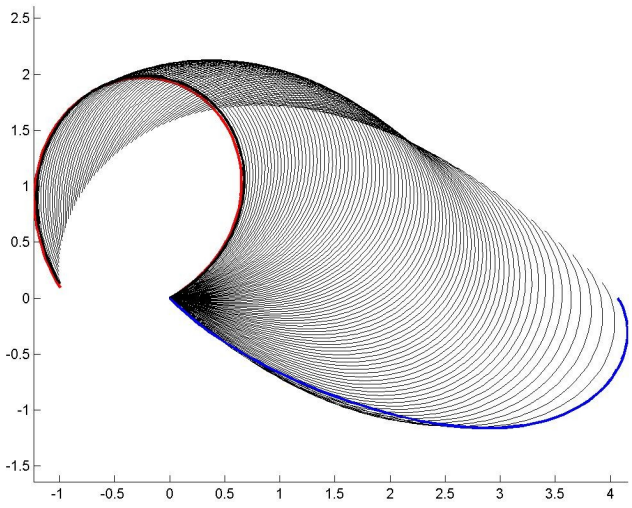

FIG. 4.3. *The movement obtained by minimizing the squared curvature problem of (2.3) with uniform rigidity coefficients.*

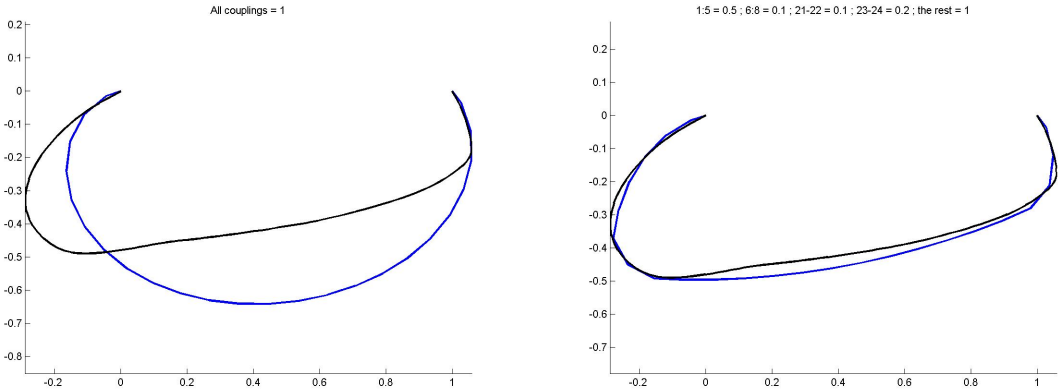

FIG. 4.4. *An example of an octopus's arm configuration, shown in black. The solution of (2.3) discretized by 24 segments, in blue, includes the case were* $a \equiv 1$ *on the left and the case with manually adjusted smaller coefficients at both ends on the right, with* $a_1 : a_5 = 0.5$, $a_6 : a_8 = 0.1$, $a_9 : a_{20} = 1$, $a_{21} : a_{22} = 0.1$, $a_{23} : a_{24} = 0.2$.

4.4. The produced extension movement. The actual movement of the octopus arm can be described as a sequence of configurations of its backbone. See for example the 3D extension movement in the left of Figure 4.5. In general, such a movement is a superposition of bend propagation and arm elongation. To create a similar movement we start from the initial configuration, the red contour in the right of Figure 4.5, i.e., from its given locations and angles of end points. Then, the consequent configurations are generated by gradual changes introduced into the initial parameters until some conditions are met for the last configuration, see the left of Figure 4.6. In addition, to achieve the bend propagation, we introduce a *shift* in the rigidity coefficients which are responsible for the bend, i.e., those in blue which create

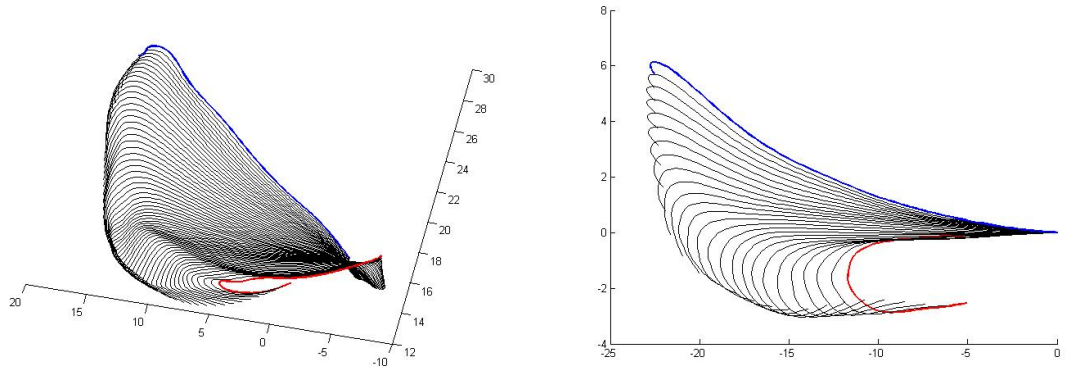

FIG. 4.5. *Left: An example of 3D octopus extension movement of bend propagation and elongation. Right: The 2D projection of that movement using principal component reduction.*

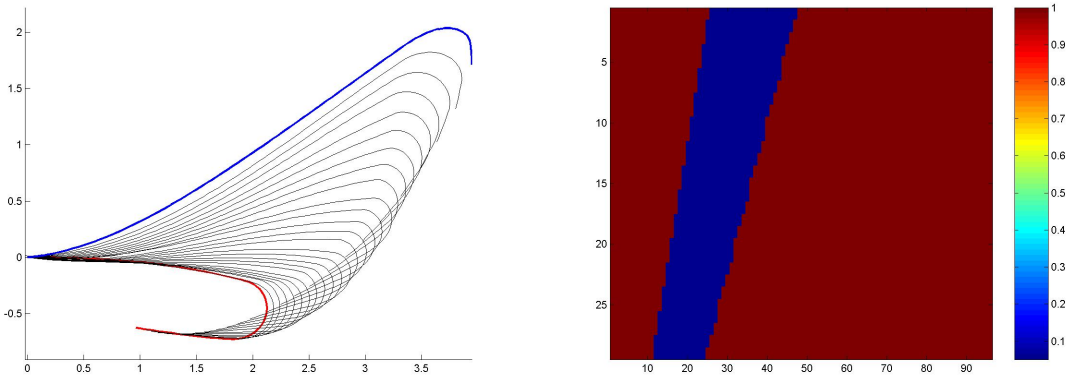

FIG. 4.6. *Left: An example of bend propagation and elongation produced by our optimization model so as to resemble the 2D octopus movement in the right of figure (4.5). Right: The rigidity coefficients we have used to obtain the bend propagation.*

the step pattern in the right of Figure 4.6.

Sometimes it is observed that the octopus starts the extension movement with an additional bend of its arm closer to its body, see the red line in Figure 4.7. In such a case it is necessary to let this bend gradually disappear as the arm becomes straight. We show here such an example which involves either only bend propagation, see the left of Figure 4.8, and one which also allows elongation of the arm, the right of Figure 4.8. In both cases the used rigidity coefficients are given in Figure 4.9.

5. Future directions. The solution techniques described here can and should serve more general problem situations (not necessarily relevant to the octopus arm, but important in many other areas), such as:

- **Using other governing functionals**: e.g., instead of $(u'(s))^2$ in (2.1), use $|u'(s)|$ (allowing unbounded curvature at some points, or generally $|u^{(m)}(s)|^q$, or combination of such terms; etc.
- Use other kinds of **boundary conditions**.

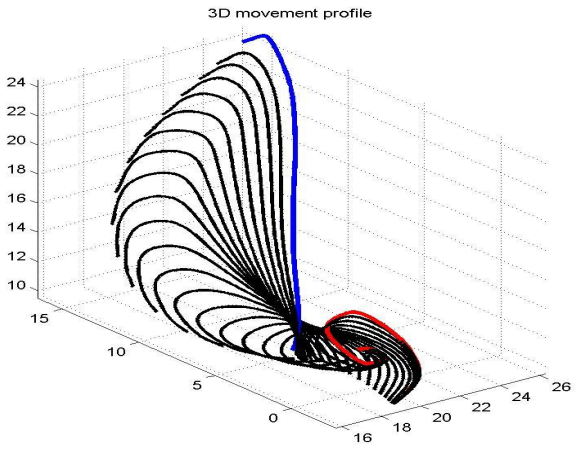

FIG. 4.7. *An example of a 3D movement of the octopus's arm.*

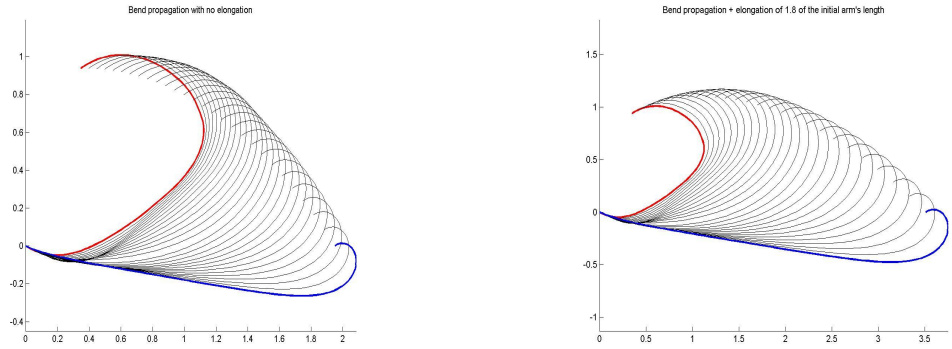

FIG. 4.8. *Left: An example of an octopus-like movement of bend propagation generated by the multigrid algorithm. Right: An example of an octopus-like movement of bend propagation combined with elongation generated by the multigrid algorithm.*

- Calculate arm **curves in 3D space**, instead of the current 2D model.
- Instead of a fixed length L, add **stiffness coefficients** (energy term associated with local stretching) to the governing functional (as those measured in [14]).
- **Dynamic arm**: simulate an arm moving in time, $u = u(s,t)$, guided by a suitable minimization functional, such as $\int E(a, u(s,t))dt$ and possibly by some stiffness coefficients.
- The results presented here were manually designed, in particular the rigidity coefficients were found by trial and error. An interesting question is to auto-

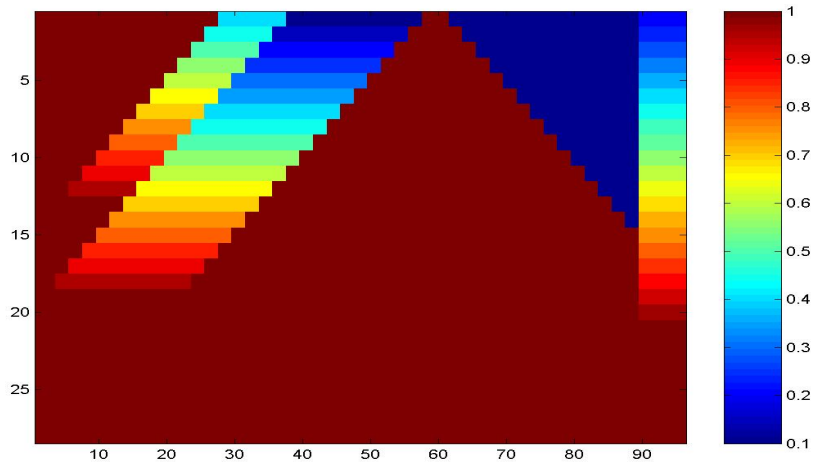

FIG. 4.9. *The rigidity coefficients used in the multigrid algorithm to create an octopus-like movement of bend propagation. The smaller coefficients (in blue and light blue step pattern) which are responsible for the produced bend, are shifted at each consequent configuration.*

matically solve the *inverse problem* in which the arm configuration, i.e., the function $u(s)$ is *given* for a few configurations (along with the length L and the boundary conditions), from which the rigidity coefficient function $a(s)$, is *derived*. (Multigrid tools for very efficient solution of inverse problems are described in [3], Section 4, [8] and in more details in [4] and [12].)

- **Dynamic inverse problem**: In the octopus case described in this paper, the rigidity coefficients actually change in time. So in such cases the inverse problem would be to calculate an unknown rigidity $a(s,t)$ so as to yield solutions $u(s,t)$ that best fit a collection of *dynamic* arm observations, regularized by adding to the governing functional terms such as

$$\int\int (\alpha[\frac{\partial}{\partial t}a(s,t)]^2 + \beta[\frac{\partial}{\partial s}a(s,t)]^2)\ ds\ dt. \tag{5.1}$$

- **System optimization**: Instead of calculating $a(s,t)$ to best fit a set of observations, seek $a(s,t)$ for which the system would operate optimally, e.g., minimizing average execution times of a set of tasks. Such a formulation is needed for example for the optimal design of *robotic* arms. (See again [3] Sections 4 and 5 with more details in [4].)
- **Stochastic formulation**: Instead of minimizing energy $E[a,u]$, assume a probability distribution P is given, for example in the Bolzmann form

$$P[a,u] = \frac{1}{z}\exp^{-\beta E[a,u]}. \tag{5.2}$$

For a given set of coefficients a, the task is the to calculate various *average properties* of the set of probable arm configurations u. In other kinds of problems (sometime falling under the name *uncertainty quantification*), the rigidity coefficients are out given either, they just satisfy some probabilistic relations, as in the Bolzmann from above. (Very efficient multiscale Markov

Chain Monte Carlo techniques for stochastic problems are described in [7] and [6]; such techniques should be combined with the multigrid methods for inverse problems to efficiently quantify uncertain diffusion and similar problems.)

Acknowledgments. This work was supported in part by the European Commission in the ICT-FET OCTOPUS Integrating Project, under contract no 231608.

REFERENCES

[1] D. Bai and A. Brandt. Local mesh refonement multilevel techniques. *SIAM J. ScI. STAT. COMPUT.*, 8(2):109–134, 1987.

[2] A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Math. Comp.*, 31(138):333–390, April 1977.

[3] A. Brandt. Multiscale scientific computation: Review 2001. In T. Barth, R. Haimes, and T. Chan, editors, *Multiscale and Multiresolution methods: Theory and Applications (Proceeding of the Yosemite Educational Symposium, October 2000)*, pages 1–96. Springer-Verlag, 2001. http://www.wisdom.weizmann.ac.il/~achi/review00.pdf.

[4] A. Brandt. Multiscale methods of data assimilation and feedback optimal control. *Technical Report*, 2006. http://www.wisdom.weizmann.ac.il/~achi/tr06-04.pdf.

[5] A. Brandt, J. Brannick, K. Kahl, and I. Livshits. Bootstrap amg. *SIAM J. SCI. COMPUT.*, 2011.

[6] A. Brandt and M. Galun. Optimal multigird algorithms for variable-coupling isotropic gaussian models. *Journal of Stat. Phys.*, 88:637–664, 1997.

[7] A. Brandt, M. Galun, and Ron D. Optimal multigird algorithms for calculating thermodynamic limits. *Journal of Stat. Phys.*, 74:313–348, 1994.

[8] A. Brandt and R. Gandlin. Multigrid for atmospheric data assimilation analysis. In T. Y. Hou and E. Tadmor, editors, *Hyperbolic Problems: Theory, Numercis, Applications*, pages 369–376. Springer, 2003.

[9] A. Brandt and O. Livne. *Multigrid Techniques: 1984 Guide with Application to Fluid Dynamics, Revised Edition.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2011.

[10] A. Brandt and D. Ron. Chapter 1 : Multigrid solvers and multilevel optimization strategies. In J. Cong and J. R. Shinnerl, editors, *Multilevel Optimization and VLSICAD*. Kluwer, 2003.

[11] A. Bruckstein and A. N. Netravale. On minimal energy trajectories. In *Journal of Comp. Vision. Graphics and Image Processing*, pages 283–296, 1990.

[12] R. Gandlin. *Ph.D. Thesis. Multigrid solvers for Inverse Problems.* PhD thesis, The Weizmann Institute of Science, 2003.

[13] Y. Gutfreund, T. Flash, Y. Yarom, G. Fiorito, I. Segev, and B. Hochner. Organization of octopus arm movements: a model system for studying the control of flexible arms. *The Journal of Neuroscience*, 16(22):7297–7307, 1996.

[14] D. Held, Y. Yekutieli, and T. Flash. Characterizing the stiffness of a multi-segment flexible arm during motion. In *IEEE International Conference on Robotics and Automation*, pages 3825–3832, 2012.

[15] B. K. Horn. The curve of least energy. *ACM Trans. Math. Soft.*, 9:441–460, 1983.

[16] M. Kallay. Plane curves of minimal energy. *ACM Transactions on Mathematical Software*, 12:219–222, 1986.

[17] Y. Kanayama and B. Haltman. Smooth local path planning for autonomous vehicles. In *Proc. Intl. Conf. on Robotics and Automation*, pages 1260–1264, Scottsdale, Arizona, 1989.

[18] W. M. Kier and K. K. Smith. Tongues, tentacles and trunks: the biomechanics movement in muscular-hydrostats. *Zoological J Linn Soc*, 83:307–324, 1985.

[19] M. Moll and L. Kavraki. Path planning for minimal energy curves of constant length. In *IEEE International Conference on Robotics and Automation*, pages 2826–2831, 2004.

[20] D. Mumford. Elastica and computer vision. In Chandrajit L. Bajaj, editor, *Algebraic Geometry and its Applications*, pages 491–506. Springer-Verlag, 1994.

[21] G. Robinson and J. B. C. Davies. Continuum robots - a state of the art. In *Proc. IEEE Conf. on Robotocs and Automation*, pages 2849–2854, Detroit, Michigan, May 1999.
[22] N. Simaan, R. Taylor, and P. Flint. A dexterous system for laryngeal surgery. *IEEE Conf. Robotics and Automation*, pages 351–357, 2004.
[23] G. Sumbre, Y. Gutfreund, G. Fiorito, and T. Flash B. Hochner. Control of octops arm extension by a peripheral motor program. *Science*, 293:1845–1848, 2001.
[24] U. Trottenberg, C. W. Oosterlee, and A. Schuller. *Multigrid*. Academic Press, Inc., Orlando, FL, USA, 2001.
[25] H. Tsukagoshi, A. Kitagawa, and M. Segawa. Active hose: an artificial elephant's nose with maneuverability for rescue operation. In *Proc. IEEE/ICRA Intl. Conf. on Robotics and Automation*, pages 2454–2459, Seoul, Korea, May 2001.
[26] A. Wolf, H. B. Brown, R. Casciola, A. Costa, M. Schwerin, E. Shamas, and H. Choset. A mobile hyper redundant mechanism for search and rescue tasks. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, volume 3, pages 2889–2895, Taipei, Taiwan, May 2003.
[27] Y. Yekutieli, R. Mitelman, B. Hochner, and T. Flash. Analysis octopus movements using three dimensional reconstruction. *J Neurophysiol*, 98:1775–1790, 2007.
[28] I. Zelman, M. Galun, A. Akselrod-Ballin, Y. Yekutieli, B. Hochner, and T. Flash. Nearly automatic motion capture system for tracking octopus arm movements in 3d space. *Journal of Neuroscience Methods*, 182:97–109, 2009.
[29] T. Zheng, D. T. Branson, R. Kang, M. Cianchetti, E. Guglielmino, M. Follador, G. A. Medrano-Cerda, I. S. Godage, and D. G. Caldwell. Dynamic continuum arm model for use with underwater robotic manipulators inspired by octopus vulgaris. In *Proc. IEEE Conf. on Robotocs and Automation (ICRA)*, pages 5289–5294, South Carolina, 2012.

Appendix A. Minimization based interpolation, the general case.

The general case considered here is the correction scheme interpolation to *many* inner fine points as is, for example, needed (in the middle of the grid) between the two coarse levels in Figure 3.1. We assume that switching to the coarse level occurred when the relaxation became slow and thus the residuals small. Since the error solves the residual equations, we may also assume that the right hand side of these equations approximately vanishes, which is equivalent to minimizing a quadratic form. To calculate the interpolation to a finer grid given a solution at a coarser level, we thus look at the following functional

$$E(a,u) = \frac{1}{2}\int a(s)(u'(s))^2 ds, \tag{A.1}$$

and its discretization, as depicted for a segment of the entire domain in Figure A.1,

$$E^h(a,u) = \frac{1}{2}\sum_{j=1}^{n-1} \overline{a}_{j+1/2}(u_{j+1}-u_j)^2, \tag{A.2}$$

where $\overline{a}_{j+1/2} = a_{j+1/2}/h_{j+1/2}$. The interpolation to the fine grid points can be calculated for the general case of variable $h_{j+1/2}$ and $a_{j+1/2}$ for $j = 1, ..., n-1$, based on *minimization* considerations. In particular, $\partial E^h(u)/\partial u_j = 0$ yields

$$\overline{a}_{j+1/2}(u_{j+1}-u_j) = \overline{a}_{j-1/2}(u_j - u_{j-1}) = \mu. \tag{A.3}$$

μ can be interpreted as a constant current flowing between the points 1 and n, while $1/a_i$ is the resistance per unit length and thus $h_{i+1/2}/a_{i+1/2} = 1/\overline{a}_{i+1/2}$ is the resistance between the points i and $i+1$. Next, we may derive from (A.3) that

$$u_n - u_1 = \mu\ \Sigma_{j=1}^{n-1}\frac{1}{\overline{a}_j}. \tag{A.4}$$

Under the assumption that the transfer to the coarse level is by *injection* (i.e., where U_K and U_M correspond to u_k and u_m, respectively, see Figure A.1 and $M = K + 1$), we may conclude that the interpolation coefficients from the two known coarser grid variables are

$$u_i \leftarrow (\uparrow_H^h U)_i = (\ U_K\ \Sigma_{j=i}^{n-1} \frac{1}{\overline{a}_{j+1/2}} +\ U_M\ \Sigma_{j=1}^{i-1} \frac{1}{\overline{a}_{j+1/2}}) \ /\ \Sigma_{j=1}^{n-1} \frac{1}{\overline{a}_{j+1/2}}, \qquad \text{(A.5)}$$

where $\uparrow_H^h$ denotes the coarse-to-fine interpolation operator. (See (3.16) in section 3.5 for the interpolation of the *correction* we actually use within the FAS.)

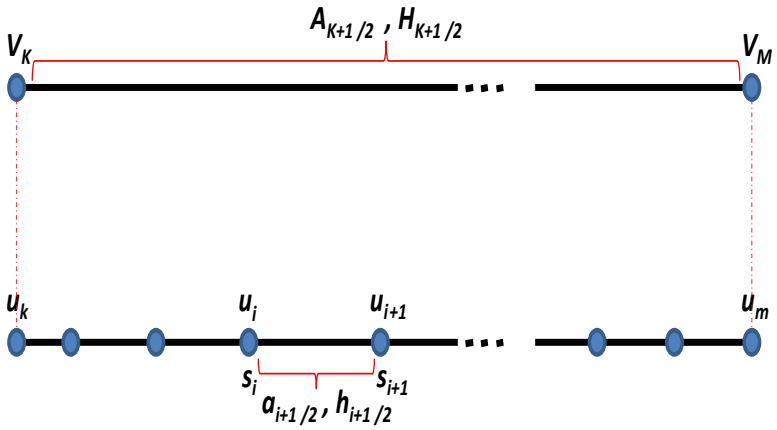

FIG. A.1. *An example of a segment of a fine grid with variable grid meshes h_i and variable coefficients a_i and its related coarse grid with only two points.*