A Vector Outline Descriptor Using Interpolatory Subdivision Curves

Carolina Beccari, Giulio Casciola and Lucia Romani

Abstract. In this paper we present an innovative vector outline description scheme that employs C^1 locally-controlled interpolatory subdivision curves to model the contours that best fit black-on-white bitmap images. The core of the method for outline extraction includes a procedure for salient features detection and an automatic algorithm for computing an appropriate set of local tensions.

§1. Introduction

Black-on-white bitmaps are made up of black and white pixels on a grid, while vector images are composed of mathematical formulas describing their contours via a family of curves that may join together either C^0 or C^1 continuously. As such, bitmaps have a fixed resolution and cannot be resized without loosing image definition, while vector graphics are resolution independent because they can be output to the highest quality at any scale. Furthermore, bitmap images tend to have much large file sizes than their vectorial counterpart and they must be often compressed to reduce information storage. For these reasons vector outlines are by far preferred in many applications of Computer Graphics, including font designing, capturing hand-drawn images on computer screens and computer-supported cartooning.

Existing vector outline descriptors employ piecewise cubic Bézier curves as the standard de facto in the representation of output images of tracing algorithms [4, 5]. The purpose of this paper is deriving an innovative subdivision-based vector outline description scheme that constitutes an efficient alternative to traditional methods.

The paper is structured as follows. We start with a brief description of the locally-controlled interpolatory 4-point scheme, then we prove its convergence to a continuously differentiable limit. Next, in Section 3 we introduce an automatic procedure for prescribing starting values of local tensions. Finally in Section 4 we propose a novel algorithm for turning bitmaps into vector outlines, that exploits the described class of subdivision curves. We conclude the paper presenting some application examples and a comparison between our model and the standard form of outlines representation through piecewise cubic Bézier curves.

§2. The Locally-Controlled Interpolatory 4-Point Scheme

Recently, Beccari et al. proposed a non-uniform and non-stationary interpolatory 4-point scheme [1] that allows to define locally the shape of the limit curve by assigning a tension value to each edge of the polyline at the coarsest refinement level. The idea of associating different tension parameters with the edges of a given control polygon was initially investigated in [3]. There it was early defined the non-uniform 4-point scheme

$$\begin{aligned} p_{2i}^k &= p_i^{k-1} \\ p_{2i+1}^k &= -w_i^k\,(p_{i-1}^{k-1}+p_{i+2}^{k-1}) + \left(\frac{1}{2}+w_i^k\right)(p_i^{k-1}+p_{i+1}^{k-1}) \end{aligned} \tag{1}$$

and it was also proved that to generate C^1-continuous limit curves the coefficients w_i^k must be chosen in the span $[\epsilon, \frac{1}{8}-\epsilon]$, with $0<\epsilon<\frac{1}{16}$. However, to make this scheme non-stationary, one still needs to find a procedure for prescribing their values at each subdivision step. The novelty of the proposal in [1] consists in finding a simple and effective formula for automatically updating the local parameters w_i^k during refinement. In particular, denoted by $P^0=\{p_i^0\}_{i\in\mathbb{Z}}$ the initial polyline and by v_i^0 the initial tension value assigned to the edge $\overline{p_i^0 p_{i+1}^0}$, at any subdivision step $k\geq 1$ the tension parameters v_i^{k-1} are updated into the new parameters v_i^k through the relation

$$v_i^k = \sqrt{\frac{1+v_i^{k-1}}{2}}\ , \tag{2}$$

and the coefficients w_i^k are computed as

$$w_i^k = \frac{1}{8v_i^k(1+v_i^k)}. \tag{3}$$

Thus, the polygon $P^{k-1}=\{p_i^{k-1}\}_{i\in\mathbb{Z}}$ is mapped into the refined polygon $P^k=\{p_i^k\}_{i\in\mathbb{Z}}$ via the refinement rules (1), where the parameters w_i^k are given by (2)-(3). Observe that, since the initial tension values v_i^0 are updated before computing the coefficients w_i^1 used in the first subdivision step, for any choice of v_i^0 in $(-1,+\infty)$, (2) implies $v_i^k>0$ and hence $w_i^k>0$ for all $k\geq 1$.

The choice of prescribing the parameters w_i^k through rules (2)-(3) turns out to be very convenient in applications where it is desirable to combine local shape control with a visually-pleasing appearance of the limit curve. In fact, firstly, it provides a very intuitive feel of the tension effect, in that, when v_i^0 tends to infinity, then w_i^1 approaches to 0, so that the portion of the limit curve confined to the endpoints of the i-th edge is pulled towards the linear interpolant of those two points. Secondly, for appropriate values of v_i^0, it guarantees reproduction of all conic sections [2]. Lastly, due to the following theorem, we can also prove that by repeating refinement rules (1)-(3) iteratively, we get a denser and denser set of data points which gives rise to a continuously differentiable limit.

Theorem 1. *The non-stationary subdivision scheme defined by (1)-(3) generates C^1-continuous limit curves for any choice of the initial tension parameters v_i^0 in the range $(-1, +\infty)$.*

Proof: By equations (2)-(3) it can be proved that, for any $v_i^0 \in (-1, +\infty)$, $w_i^k \to \frac{1}{16}$ when $k \to \infty$ [1]. This implies $w_i^k \in [\epsilon, \frac{1}{8} - \epsilon]$ $(0 < \epsilon < \frac{1}{16})$ and then from Theorem 3.1 in [3] the proposed non-uniform 4-point scheme converges to a C^1-continuous limit curve. $\square$

§3. Automatic Setting of Local Tensions

Local tensions constitute a powerful tool increasing the attractiveness of subdivision in applications. Now, in order to make the most of it, the outstanding issue is: how local tension values can be opportunely set up? It is clear that, given a starting control polygon, a one-way choice of tensions does not exist. Deciding which might be the most plausible one strongly depends on specific requirements. In [1] and [2] a strategy to assign starting tension values v_i^0 ensuring in the limit an optimal fitting of piecewise cubic Bèzier curves and the exact reconstruction of an arbitrary conic section was respectively described. In the following we will illustrate a more general procedure that is capable to work out values of local tension which can help in many practical cases. The method requires as input two sets of data. The first is simply the polyline $P^0 = \{p_i^0\}_{i\in\mathbb{Z}}$ at the coarsest level. The second is a collection of *target points* $T = \{t_i\}_{i\in\mathbb{Z}}$, that acts as a hint for describing the shape of the limit curve we want to generate. In this way each t_i is meant to be associated with the edge $\overline{p_i^0 p_{i+1}^0}$ and it represents the point to be inserted by the scheme during the first refinement. But, since the classical 4-point insertion rule applied to p_{i-1}^0, p_i^0, p_{i+1}^0, p_{i+2}^0 defines a new point on the line passing through the mid-points of the edges $\overline{p_i^0 p_{i+1}^0}$ and $\overline{p_{i-1}^0 p_{i+2}^0}$, in order to make each t_i to be surely reached by the first refinement on P^0, we insert in (1) an additional parameter $\lambda_i \in \mathbb{R}$, so that

$$p_{2i+1}^1 = \frac{p_i^0 + p_{i+1}^0}{2} + 2w_i^1 \left\{ \frac{p_i^0 + p_{i+1}^0}{2} - \left[(1-\lambda_i)p_{i-1}^0 + \lambda_i p_{i+2}^0\right] \right\}. \quad (4)$$

Hence, by solving the linear system of equations deriving by imposing componentwise $t_i = p_{2i+1}^1$, we get

$$\lambda_i = \frac{\overline{y}(x_i+x_{i+1}-2x_{i-1})+x_{i-1}(y_i+y_{i+1})-\overline{x}(y_i+y_{i+1}-2y_{i-1})-y_{i-1}(x_i+x_{i+1})}{(y_{i+2}-y_{i-1})(x_i+x_{i+1}-2\overline{x})+(x_{i+2}-x_{i-1})(2\overline{y}-y_i-y_{i+1})}$$

(5)

and

$$w_i^1 = \frac{1}{2}\left|\frac{2\overline{x} - x_i - x_{i+1}}{x_i + x_{i+1} - 2x_{i-1} + 2\lambda_i(x_{i-1} - x_{i+2})}\right| , \tag{6}$$

where $(\overline{x}, \overline{y})$ and (x_j, y_j), $j = i-1, i, i+1, i+2$ denote the coordinates of points t_i and p_j^0 respectively.
Note that, when $\lambda_i = \frac{1}{2}$, equation (4) coincides with the second in (1). In this situation the direction of insertion is the one passing through the mid-points of the edges $\overline{p_i^0 p_{i+1}^0}$ and $\overline{p_{i-1}^0 p_{i+2}^0}$ and the coefficient w_i^1 indicates how far from the edge $\overline{p_i^0 p_{i+1}^0}$ the inserted point t_i should be moved. In the very general case the two insertion rules do not coincide, but w_i^1 is still closely related to the distance of t_i from the edge $\overline{p_i^0 p_{i+1}^0}$. Indeed, the value of w_i^1 still suggests how much the modulus of the displacement vector needs to be scaled to reach the desired target point.
Next, the tension value v_i^1 can be worked out from the coefficient w_i^1 computed in (6), by inverting equation (3) and choosing the positive square root, i.e.

$$v_i^1 = \frac{\sqrt{2w_i^1(2w_i^1 + 1)}}{4w_i^1} - \frac{1}{2} . \tag{7}$$

In short the general algorithm to compute the local tension v_i^1 to be associated with the edge $\overline{p_i^0 p_{i+1}^0}$ can be outlined as follows.

Algorithm 1. : local tension parameter computation

```
Input: the quadruple of points $p_j^0 = (x_j, y_j)$, $j = i-1, i, i+1, i+2$ and
       the target point $t_i = (\overline{x}, \overline{y})$.
 1. Compute the value of $\lambda_i$ through (5).
 2. Work out the value of $w_i^1$ given by (6).
 3. Determine $v_i^1$ using (7).
Output: the local tension value $v_i^1$ to be associated with the edge $\overline{p_i^0 p_{i+1}^0}$.
```

In the following such an approach will be employed to automatically set the initial tension values to be placed in the rules (1)-(3) for defining the locally-controlled subdivision curves that best approximate the contours of a black-on-white bitmap image.

§4. The Subdivision-Based Tracing Algorithm

In this section we present the procedure that, starting from a black-on-white bitmap image, allows us to extract its contours and describe them by means of interpolatory subdivision curves. Such a process, technically named *tracing*, can be achieved through the following algorithm.

Algorithm 2. : subdivision-based tracing

I. *Reference polygon generation.*
A set of closed paths, forming the boundaries between black and white areas, is extracted from the bitmap and for each of them a polygon Q, which approximates its shape, is defined.

II. *Coarsest level definition.*
A starting polygon P^0 is worked out from each reference polygon Q and, for all its edges, a target point and an associated initial tension parameter are determined, in such a way refining the set of all starting polygons through the rules (1)-(3), a number of smooth scalable vector outlines approximating the bitmap are generated.

Among all available methods that could be used to efficiently perform computations in phase I, in our implementation we have exploited the one presented in [5]. As a consequence, we will not treat this phase in detail, but we will confine ourselves to emphasize only the changes we carried out to adjust the adopted procedure to fit specific requirements related to subdivision. Instead, we will handle phase II extensively.

4.1. Reference polygon generation (phase I)

In this phase we extract from the bitmap a set of closed paths that captures its contours. Then, from each of them, we define a polygon Q, that will be called *reference polygon*, which approximates the shape described by the path vertices. Out of all possible polygons that fulfill this requirement, we will select the one that possesses:

- the minimum number of edges (in order to minimize the descriptor size and reduce storage and computational costs of the vector image);
- the minimum euclidean distance between edges and path vertices (in order to improve approximation quality of the outlines);
- the minimum difference in length between edges (in order to avoid artifacts generation in the outlines).

The above criterions are examined in order of importance, i.e., the first will be considered prior to the second and the second prior to the third. Note that, since interpolatory 4-point schemes can generate unpleasant artifacts if applied to non-equidistant data, to obtain nice-looking limit curves we will select polylines with vertices that are the most uniformly spaced as possible. However, this can be put as last requirement in order of importance, since, by a wide range of experiments, there is an evidence that the first two conditions look sufficient to guarantee visually pleasing outline curves. This happens because, as we will see later, the shape of the limit curves is driven by the constraint to fit opportunely chosen target points, lying close to the original bitmap.

4.2. Coarsest level definition (phase II)

In this subsection we will see how to determine a set of polygons at the coarsest level of detail, in such a way that, after refinement, a number of smooth outline curves closely approximating the original bitmap are produced. Every *starting polygon*, here denoted by P^0, can be derived from the reference polygon Q, after having applied the *salient features detection* procedure described in Algorithm 3. The vertices of the starting polygon P^0 will be defined by the sequence of mid-points of the edges in Q, completed by the insertion of all corners q_i detected as follows.

Algorithm 3. : salient features detection

```
1. Consider three subsequent vertices $q_{i-1}, q_i, q_{i+1}$ of the
   reference polygon $Q$ derived in phase I.
2. Define by $R$ and $S$ the triangles with vertices $q_{i-1}, q_i, q_{i+1}$
   and $q_{i-1}, s, q_{i+1}$ respectively, where $s$ is a point on the
   perpendicular to $\overline{q_{i-1}q_{i+1}}$ such that $\overline{sq_{i-1}} = 1$.
3. Define $A_i = \frac{area(R)}{area(S)}$
4. Compare $A_i$ with $A_{max}$:
   4.1. if $A_i > A_{max}$, then mark $q_i$ as corner
   4.2. else associate with $q_i$ the following parameter value $\gamma_i$:
        4.2.1. if $A_i \leq 1 + \frac{\sqrt{2}}{2}$, then $\gamma_i = \frac{1}{\sqrt{2}+1}$
        4.2.2. if $1 + \frac{\sqrt{2}}{2} < A_i < 4$, then $\gamma_i = 1 - \frac{1}{A_i}$
        4.2.3. if $4 \leq A_i \leq A_{max}$, then $\gamma_i = \frac{3}{4}$
```

The algorithm above exploits the parameter A_{max} as a threshold to determine whether a corner has to be created or not in the limit curve. Such a parameter can be set to any arbitrary value in $[4, +\infty)$: the greater the value of A_{max}, the smaller the number of detected corners. This value has to be assigned before starting the tracing procedure and it must be the same for each run of Algorithm 3 all over the reference vertices q_i. Condition $A_i > A_{max}$ indicates the presence of a corner, i.e. a point where the

limit curve should be C^0. Conversely, condition $A_i \leq A_{max}$ means that the limit curve exhibits a nice smooth behaviour. Thus, we will handle these two situations separately.
Let us start by analyzing the case $A_i \leq A_{max}$. Once the features detection procedure has been applied, a parameter value γ_i is assigned to each reference vertex q_i that does not correspond to a corner. The value of γ_i determined by Algorithm 3 clearly depends on the relative position of the line $\overline{q_{i-1}q_{i+1}}$ with respect to the pixel centered at q_i. Such a parameter is used to define the target point t_i as follows:

$$t_i = \frac{p_i^0 + p_{i+1}^0}{2} + \gamma_i \left(q_i - \frac{p_i^0 + p_{i+1}^0}{2} \right). \tag{8}$$

Equation (8) clearly states that t_i will be placed along the line connecting the mid-point of the edge $\overline{p_i^0 p_{i+1}^0}$ and the point q_i, in a position that depends on the computed γ_i: the larger the value of γ_i, the closer the point t_i will be to q_i. The value of γ_i that fulfills our requirements has been derived by separating three different significant situations. This distinction translates into checking to which of the three spans defined in step 4.2 of Algorithm 3 the value of A_i belongs.

- The first situation depicts a limit curve that does not stray far from the edge $\overline{p_i^0 p_{i+1}^0}$. Thus, to prevent it from becoming too flat, we can conveniently set the lower bound on γ_i to the value $\frac{1}{\sqrt{2}+1}$, which corresponds to a quarter circle reproduction. In fact, as it was proved in [2], if we sample four equally-spaced points $p_{i-1}^0, p_i^0, p_{i+1}^0, p_{i+2}^0$ on a circle, the 4-point scheme (1)-(3) is able to reproduce a 45 degrees circular arc in correspondence to the edge $\overline{p_i^0 p_{i+1}^0}$, whenever we assign the value 0 to the tension parameter v_i^0. This implies $v_i^1 = \frac{\sqrt{2}}{2}$ and $w_i^1 = \frac{1}{4(\sqrt{2}+1)}$. Moreover, since the target point t_i will be defined by the refinement rule in the second line of (1) as

$$t_i = \frac{p_i^0 + p_{i+1}^0}{2} + 2w_i^1 \left(\frac{p_i^0 + p_{i+1}^0}{2} - \frac{p_{i-1}^0 + p_{i+2}^0}{2} \right), \tag{9}$$

if we denote the expression in brackets by $\ell\vec{u}$, from equation (8) it can be easily seen that, to place a new point on the circle arc, it is necessary to assume $q_i - \frac{p_i^0 + p_{i+1}^0}{2} = \frac{1}{2}\ell\vec{u}$. Next, by equating relations (8) and (9), it trivially turns out that the value of γ_i for a quarter circle reproduction is exactly $\frac{1}{\sqrt{2}+1}$. This leads to $A_i \leq 1 + \frac{\sqrt{2}}{2}$.

- The second situation corresponds to a limit curve approaching the vertex q_i. Since we do not want it to assume a “spiky” shape, the upper bound $\gamma_i \leq \frac{3}{4}$ has been imposed. Such a condition translates into $A_{max} = 4$ and hence this value can be assumed as default threshold parameter for corner detection.

- The last situation is that of a limit curve at a midway position between the edge $\overline{p_i^0 p_{i+1}^0}$ and the vertex q_i. Thus we would like its shape to vary continuously between the imposed bounds, in a way that is directly proportional to A_i. For this reason we will assume $\gamma_i = 1 - \frac{1}{A_i}$, which implies that A_i spans the interval $[1 + \frac{\sqrt{2}}{2}, 4]$.

The computed values of γ_i allow us to derive the set of target points t_i needed to start the automatic procedure described in Section 3 and work out the local tensions v_i^1 to be associated with the starting polygon P_0. As concerns the case $A_i > A_{max}$, the target point to be chosen in correspondence of each vertex q_i, previously marked as corner, is instead $t_i = \frac{p_i^0 + p_{i+1}^0}{2}$. This choice is related to a "high" value of tension that we must assign to the parameters v_{i-1}^1 and v_i^1 associated with the two edges meeting at q_i. Additionally, in order to obtain a C^0 behaviour of the limit curve at the corner, the starting polyline P^0 will be assumed to be open at q_i and hence refined by exploiting the linear extrapolatory rule in [2].

In this way, by recursively applying the interpolatory scheme (1)-(3) with local tensions v_i^1 to the collection of starting polygons $P^1 = P^0 \cup T$, we are able to generate a set of smooth outlines made of subdivision curves joined together either C^0 or C^1 continuously, that guarantees reproduction of all salient features in the original bitmap (see Fig.1).

§5. Conclusions

In this paper we have presented a new algorithm for bitmap-vector conversion, based on a C^1 locally-controlled interpolatory subdivision scheme. The following figures illustrate the vector outlines generated by running

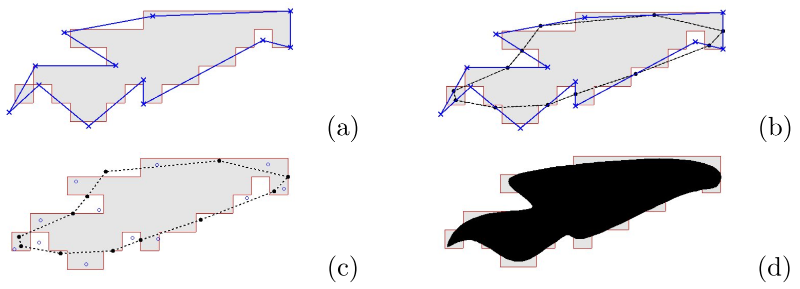

Fig. 1. Magnified view of Einstein's eyebrow (see Fig.2): (a) reference polygon, (b) reference (solid line) and starting (dashed line) polygons, (c) starting polygon and target points, (d) filled outline generated after refinement.

the proposed procedure on the bitmaps we got from a digitalized hand-drawn image (Fig.2) and from a character produced through Metafont (Fig.3). In Fig.3 a comparison with the traditional piecewise cubic Bézier representation (realized through the software in [5]) is also available.
As it appears and it was confirmed by a wide range of experiments, the novel subdivision-based tracing algorithm is able to provide an outline description of reduced size, with respect to a standard Bézier representation of the same quality.

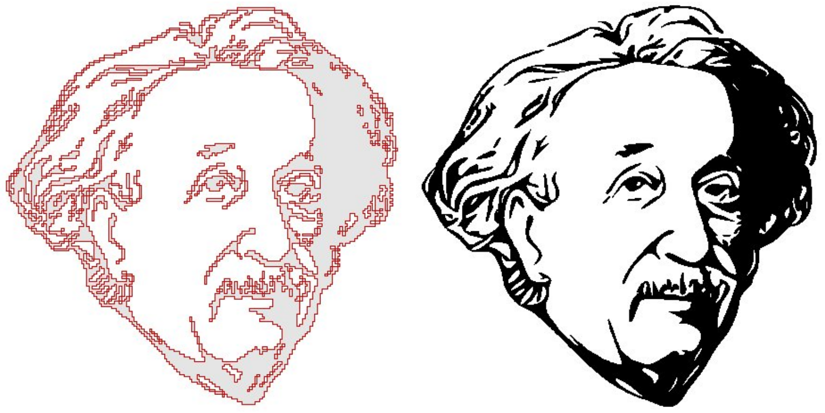

Fig. 2. Vector image obtained by applying the subdivision-based tracing algorithm to a bitmap made of 179×172 pixels depicting Einstein's face.

Acknowledgments. This work was supported by University of Bologna "Funds for selected research topics".

References

1. Beccari, C., Casciola, G., Romani, L., Interpolatory subdivision curves with local shape control, in *WSCG'2006 Full Papers Conference Proceedings*, 33–40, `http://wscg.zcu.cz/DL/wscg_DL.htm`.
2. Beccari, C., Casciola, G., Romani, L., A non-stationary uniform tension controlled interpolating 4-point scheme reproducing conics. Comput. Aided Geom. Design **24**(1) (2007), 1–9.
3. Levin, D., Using laurent polynomial representation for the analysis of non-uniform binary subdivision schemes. Advances in Comp. Math. **11** (1999), 41–54.
4. Sarfraz, M., Khan, M.A., An automatic algorithm for approximating boundary of bitmap characters. Future Generation Computer Systems **20**(8) (2004), 1327–1336.

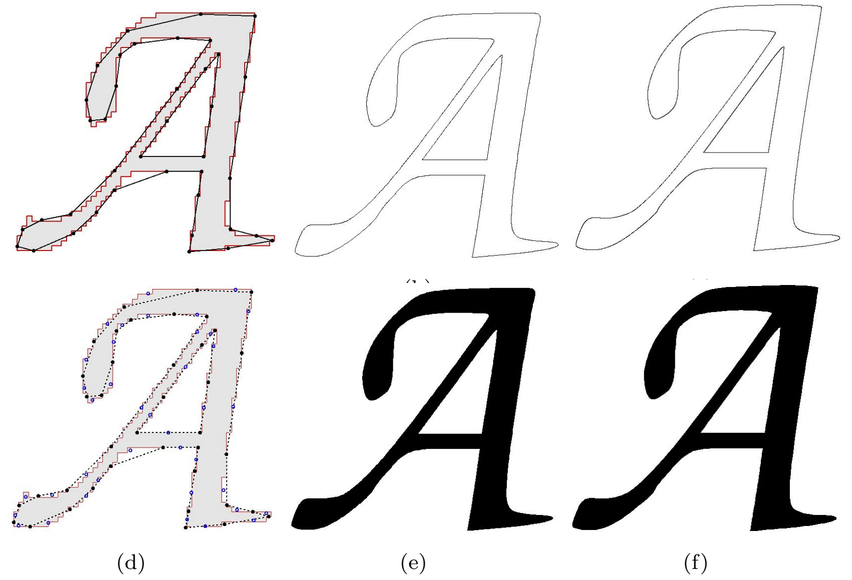

Fig. 3. Vector outline obtained from a Metafont character representing letter "A". Left column: the original bitmap (68×64 pixels) with two starting polygons made of 41 vertices (a) and the corresponding 41 target points (d). Central column: limit curves obtained through the proposed subdivision scheme (b) and corresponding filled outline (e). Right column: piecewise cubic Bézier curves with 119 control points (c) and corresponding filled outline (f).

5. Selinger, P., Potrace: a polygon-based tracing algorithm, manuscript, September 2003, `http://potrace.sourceforge.net/potrace.pdf`.

Carolina Beccari
Dept. of Pure and Applied Mathematics - University of Padova
Via G. Belzoni 7, 35131 Padova, Italy
`beccari@dm.unibo.it`

Giulio Casciola and Lucia Romani
Dept. of Mathematics - University of Bologna
P.zza Porta San Donato 5, 40127 Bologna, Italy
`casciola@dm.unibo.it`, `romani@dm.unibo.it`