`OptimalInpaiting` parallel software for image inpainting via sub-Riemannian minimizers on the group of rototranslations*

Alexey P. Mashtakov and Yuri L. Sachkov

Program Systems Institute, Pereslavl-Zalessky, Russia,
E-mail: alexey.mashtakov@gmail.com, sachkov@sys.botik.ru

Abstract. The paper is devoted to a parallel software for image inpainting developed on the basis of neurogeometry of vision and sub-Riemannian geometry. A parallel algorithm and software to restore monochrome binary or halftone images represented as series of isophotes (level lines of brightness) were developed. The method of inpainting is based on a variational principle: the recovered isophote should have a minimum length in the space of contact elements.

Keywords: Image inpainting, sub-Riemannian geometry, neurogeometry of vision, parallel software

1 Image inpainting, neurogeometry of vision, and sub-Riemannian geometry on the group of rototranslations of a plane

The paper is devoted to a parallel software for image inpainting developed on the basis of provisions of a new direction of neuroscience — neurogeometry [2, 3], as well as recent results on sub-Riemannian geometry [4–6]. On the basis of these studies were developed an algorithm and a software to restore monochrome binary or halftone images represented as series of isophotes (level lines of brightness). The mathematical foundation and the algorithm were described in works [4–7]. In this paper we present the corresponding set of parallel software `OptimalInpaiting` for reconstruction of corrupted images.

The following assumptions are adopted in development of the software:

- analytical data on corrupted image are known to the user,
- original image can be represented by a family of level lines of a smooth function without critical points,
- corrupted domains of the image are disks.

* Supported by Russian Foundation for Basic Research, Project No. 09-01-00246-a, and by The SKIF Supercomputer Project of the Russia-Belarus Union State [1]

Mathematically, our method of inpainting is based on the following variational principle: the recovered arc (isophote of the image) should have a minimum length in the space of contact elements (x, y, θ), $\theta = \arctan dy/dx$:

$$\int \sqrt{\dot{x}^2 + \dot{y}^2 + \alpha^2 \dot{\theta}^2}\, dt \to \min, \qquad \alpha = \text{const} > 0, \tag{1}$$

where the parameter α reflects the weight of the linear and angular velocities in the integral compromise (1) minimized. Another approach to inpainting via sub-Riemannian geometry on SE(2) is taken in works [8, 9].

We consider the problem of recovering a monochrome (binary of gray-scale) image, some fragments of which are corrupted or hidden from observation. The goal is to restore the corrupted parts of the image in an anthropomorphic (natural for a human being) way. The problem can be formalized as follows. Given a domain $D \subset \mathbb{R}^2$, mutually disjoint subdomains $O_1, ..., O_N \subset D$, and a function $f : D \setminus (\bigcup_{i=1}^{N} O_i) \to [0, 1]$, one should restore the function f in the domains $O_1, \ldots, O_N$. Here D is the domain of the initial image, O_i are subdomains with corrupted image, and the function f determines the image (brightness for gray-scale image, and for binary image it is a function, whose level lines coincide with the curves constituting the image). We propose to restore the in subdomains O_i by completing isophotes — level curves of f in these subdomains (in the case of halftone images, the strips between the reconstructed curves are painted according to the brightness values on these curves). The reconstructing curves are calculated via the variational approach (1).

Problem (1) is formalized as the following optimal control problem [10]:

$$\dot{x} = u_1 \cos\theta, \quad \dot{y} = u_1 \sin\theta, \quad \dot{\theta} = u_2, \tag{2}$$

$$q = (x, y, \theta) \in M = \mathbb{R}^2_{x,y} \times S^1_\theta, \quad u = (u_1, u_2) \in \mathbb{R}^2, \tag{3}$$

$$q(0) = q_0 = (0, 0, 0), \qquad q(t_1) = q_1 = (x_1, y_1, \theta_1), \tag{4}$$

$$l = \int_0^{t_1} \sqrt{u_1^2 + \alpha^2 u_2^2}\, dt \to \min. \tag{5}$$

The state space of this problem $M = \mathbb{R}^2_{x,y} \times S^1_\theta$ is naturally identified with the group SE(2) of orientation-preserving motions (rototranslations) of a two-dimensional plane. Then problem (2)–(5) is obviously reformulated as a left-invariant sub-Riemannian problem on the Lie group SE(2) [10].

In works [4–7], finding sub-Riemannian minimizers for problem (2)–(5) was reduced to solving systems of 3 algebraic equations in 3 elliptic functions in certain domains. In this paper we describe a parallel software `OptimalInpaiting` developed for reconstruction of images via the variational approach (1).

2 `OptimalInpaiting` : Logical structure

The structure of `OptimalInpaiting` software is shown in Fig. 1. A user works with the software through an interface. First, the interface creates the original

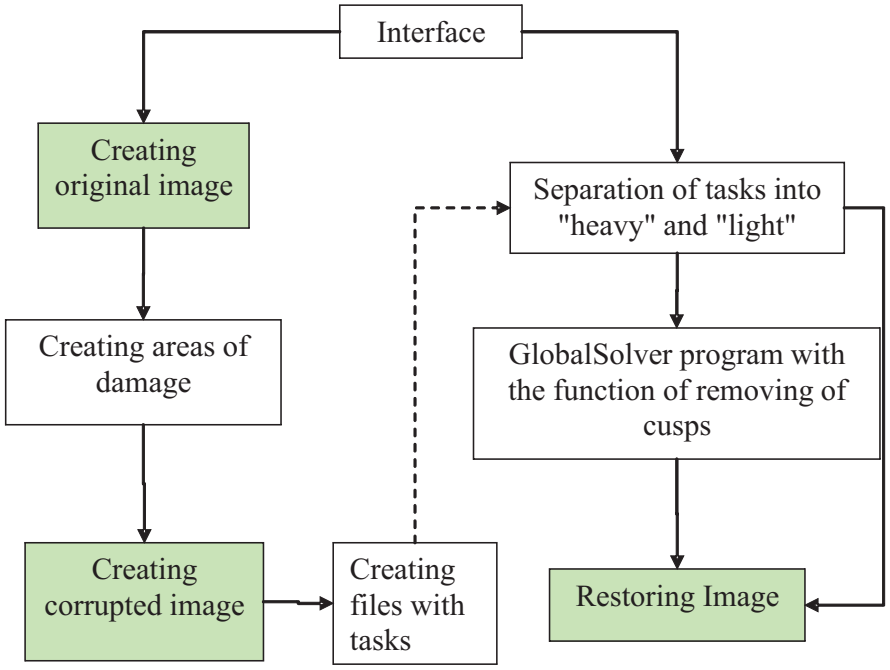

Fig. 1. Structure of `OptimalInpaiting`

image, and then the corrupted subdomains for the image. After that the corrupted subdomains are applied to image. The result is a corrupted image in accordance with the parameters chosen by the user. Then tasks are being created, each of which corresponds to a corrupted isophote (level line of brightness) in corrupted subdomains. Parameters of the tasks are calculated on the basis of information about corrupted subdomains. Tasks are divided into "heavy" and "light": the task is considered light if the boundary conditions define a straight-line isophote (up to a user-specified threshold), otherwise the task is considered heavy. Straight lines are taken as a solution of light tasks. The boundary conditions for heavy tasks are split into files corresponding to the corrupted subdomains. Each problem is solved as an optimal control problem, which was reduced to solving systems of algebraic equations [4–7]. The systems of equations are solved numerically using GlobalSolver software [7]. The solution curve corresponding to the parameters found may have cusps (i.e., be not smooth), see Fig. 3 [7]. To eliminate the non-smoothness, the parameter α is used, see (1). The structure of GlobalSolver and the use of the parameter α is described in [7]. The output of the program GlobalSolver is an output file, which lists the values of the parameters that define the optimal curves for the respective tasks. Ac-

cording to the obtained values, recovering isophotes are constructed and applied to the corrupted image. As a result, the output of `OptimalInpaiting` obtains the restored image.

2.1 Input parameters

User interaction with the application is performed via a graphical interface designed in language Tcl/Tk. Fig. 2 shows the screenshot of input form of the interface that specifies the parameters required for the application. All input

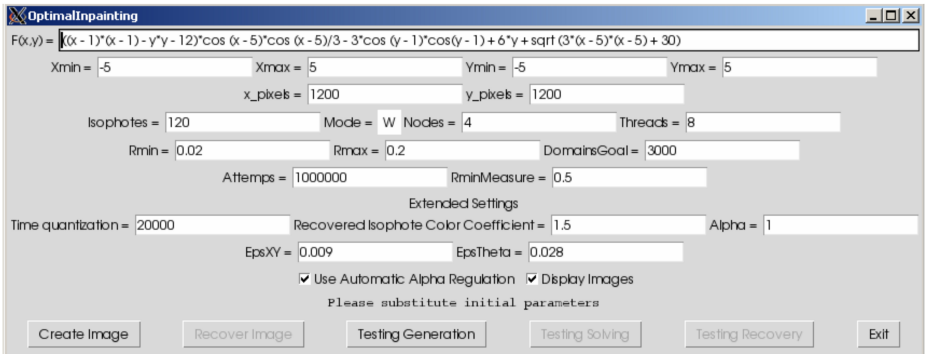

Fig. 2. Input form

data are logically divided into 4 groups:

1. Defining the original image (`F(x, y)`, `Xmin`, `Xmax`, `Ymin`, `Ymax`, `x_pixels`, `y_pixels`, `Isophotes`, `Mode`, `Threads`),
2. Defining the process of calculating the parameters of curves recovering isophotes (`Nodes`, `Alpha`, `EpsXY`, `EpsTheta`, `UseAutomaticAlphaRegulation`),
3. Defining corrupted subdomains (`Rmin`, `Rmax`, `DomainsGoal`, `Attempts`, `RminMeasure`),
4. Advanced (`Time_quantization`, `Recovered_Isophote_Color_Coefficient`, `DisplayImages`).

2.2 Process of executing

When all input parameters are entered, the user should click `CreateImage`. This creates a text file *fcutcurves.cpp*, in which the problem is written in C++ language: it includes definition of the function `F(x, y)` and parameterization of closed curves bounding the corrupted subdomains. Further work of the application is illustrated in Fig. 3.

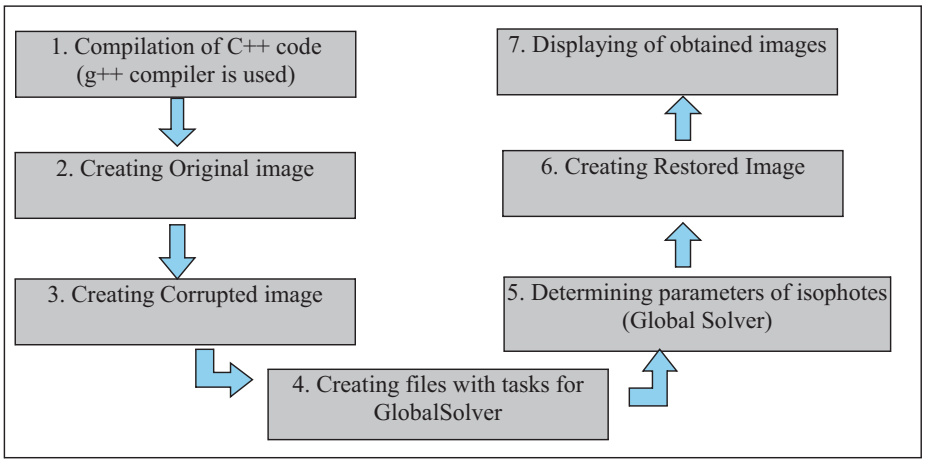

Fig. 3. Process of executing `OptimalInpaiting`

2.3 Output

As a result, `OptimalInpaiting` displays 6 windows that contain:

1. the original image (Original),
2. the corrupted image (Corrupted),
3. the corrupted image with traced boundaries of the corrupted subdomains (Corrupted Boundary),
4. the restored image (Restored),
5. the restored image with traced boundaries of the corrupted subdomains (Restored Boundary),
6. statistics collected during execution.

In the case of testing (buttons `TestingGeneration` and `TestingRecovery`), the plots demonstrating efficiency of parallelization are additionally displayed (plot of time and plot of acceleration). Figure 4 shows a screenshot of `OptimalInpaiting` application.

3 `OptimalInpaiting` : User's description

3.1 Technical Information

`OptimalInpaiting` was developed in Linux environment (Alt Linux, kernel 2.6.27-hpc-std-alt2) with the compiler gcc-4.4. The Graphical User Interface (GUI) was written in Tcl/Tk interpreted language. Tcl/Tk program is executed with the use of wish interpreter. The computational module was written in C++ with the use of tsim, libgomp, libgsl, libpng libraries. Images in pgm format are created with the use of Potrace utility.

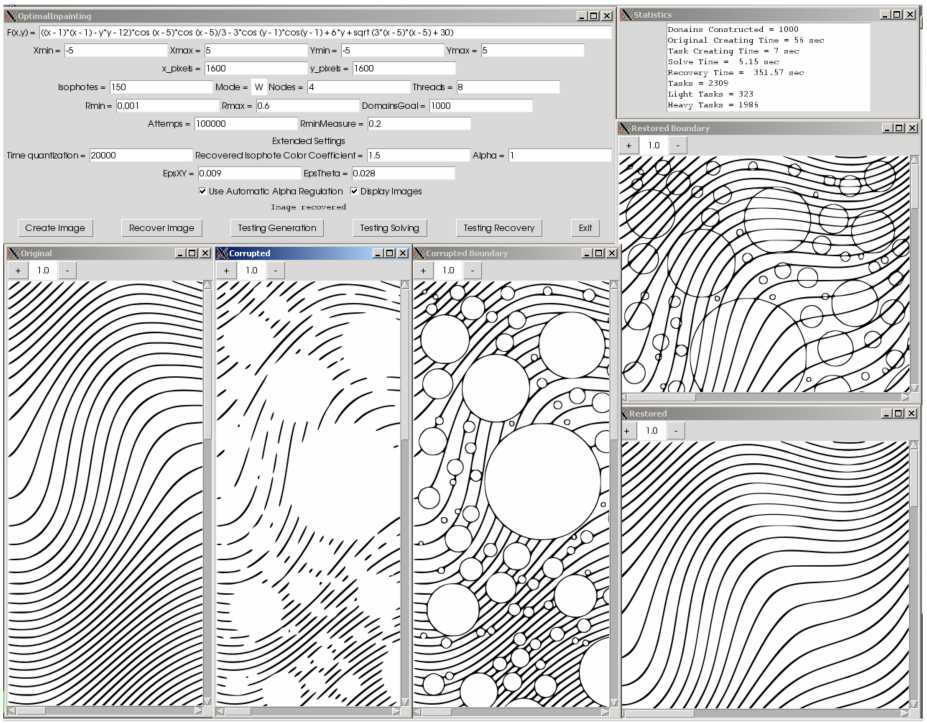

Fig. 4. `OptimalInpaiting` application

Since the computational module uses algorithms for parallel computing, it is recommended to run `OptimalInpaiting` on high performance cluster systems in order to reduce execution time.

`OptimalInpaiting` is installed on a server running under Linux. User interaction with the application occurs via GUI. Connection to the server is implemented on SSH. It is necessary to have a running X server on user's computer. To work with the PC under Windows it is suggested to use XMing and Putty.

3.2 Starting `OptimalInpaiting`

On the client machine, with access to Internet, a user should launch X-terminal and install ssh-connection with permitted X-forwarding.

To install `OptimalInpaiting` , unzip the file with the application:
`unzip -o OPTIMALINPAINTING.zip`
Then go to the working directory and configure the system:

```
cd ~/OPTIMALINPAINTING
chmod 755 configure
./configure
```

To start GUI, execute the following command:
`wish imagerecovView`
This command initiates `OptimalInpaiting` and displays the input form (Fig. 2).

3.3 Formulation and solution of the problem of image restoration using `OptimalInpaiting`

In the field `F(x, y)` the user should enter a smooth function of two arguments (the level curves of this function determine the original image). The function `F(x, y)` is written in C-style (in accordance with math.h), i.e.:

- The basic arithmetic operations: $+$, $-$, $*$, $/$,
- x to the power n: `pow(x, n)`,
- Trigonometric functions: `sin()`, `cos()`, `tan()`,
- Inverse trigonometric function: `acos()`, `asin()`, `atan()`,
- Exponential and natural logarithm: `exp()`, `log()`,
- Hyperbolic functions: `cosh()`, `sinh()`, `tanh()`.

After entering a correct input, the user can launch the solution process. To do this, click the button `Create Image` in the bottom of the Input form. This starts the process of creating original and corrupted images. The images obtained are displayed on the screen. To start the recovery process, press button `RecoveryImage`. The problem will be solved on the number of nodes prescribed by the used in the field `Nodes` (0 means execution in sequential mode). As a result, the program displays 6 windows, see (4), which contain the original image, the corrupted image, the corrupted image with traced boundaries of the corrupted subdomains, the restored image, the restored image with traced boundaries of the corrupted subdomains, and statistics. Statement of the problem of image recovery is saved in a file *fcutcurves.cpp*. It contains notation of functions that define the contours of the corrupted subdomains in the parametric form (parameter t), the number of corrupted subdomains, the initial and final value of t for each contour, the function `F(x, y)` (whose level lines determine the original image), the coefficient of smoothing for binary images and number of discretization steps of t. The images obtained are stored in directory `\pict` in two formats (png and pgm). The file *Statistic.txt* contains collected statistics about process of execution of the application.

3.4 Effectiveness of parallelizing

PC `OptimalInpaiting` includes parallel computation at 3 stages: creating of original image, determining the parameters of isophotes (GlobalSolver), creating of recovered image. TSim and OMP are used for organization of parallel computation. The application includes functions for testing effectiveness of parallelizing. To start the testing process of creating original and corrupted images with different numbers of threads (`Threads`), click `TestingGeneration`. To start testing the module determining the parameters of isophotes for different

numbers of nodes (`Nodes`), click `TestingSolving`. To start the testing process of creating recovered image with different numbers of threads (`Threads`), click `TestingRecovery`. Figs. 11, 12 show examples of testing of effectiveness of parallelizing.

4 `OptimalInpaiting` : Presentation of some results

In this section we present results of output of `OptimalInpaiting` software for the test problem described as follows:

$$F(x,y) = 1.5(x\cos y\cos y\sin x\sin x + y\sin y\sin y\cos x\cos x) \\ +x^2 - y^2 - xy - x + 2y, \tag{6}$$

$$\texttt{Xmin} = -10, \quad \texttt{Xmax} = 10, \quad \texttt{Ymin} = -10, \quad \texttt{Ymax} = 10, \tag{7}$$

$$\texttt{x\_pixels} = \texttt{y\_pixels} = 2500, \quad \texttt{Isophotes} = 100, \tag{8}$$

$$\texttt{Rmin} = 0.05, \quad \texttt{Rmax} = 0.4, \quad \texttt{DomainsGoal} = 10^4, \quad \texttt{Attempts} = 10^6, \tag{9}$$

$$\texttt{Nodes} = 4, \quad \texttt{Threads} = 8. \tag{10}$$

Figures 5–7 present respectively the original, corrupted, and restored images for problem (6)–(10) in the halftone version ($\texttt{Mode} = G$), and Figures 8–10 present the same in the binary version ($\texttt{Mode} = W$). Plots of time and acceleration vs number of threads for the process of creating of images are shown respectively in Fig. 11 and Fig. 12.

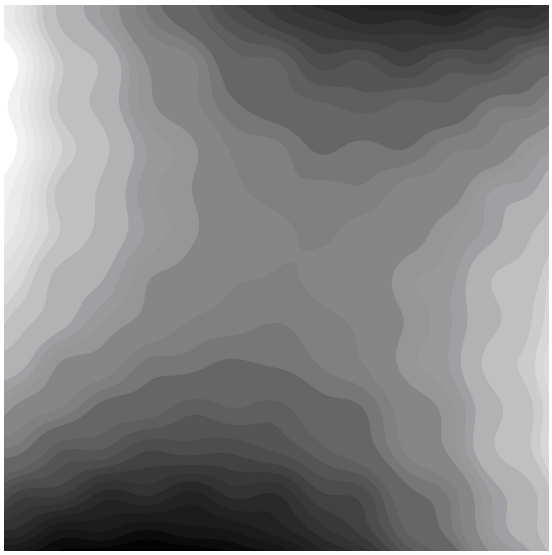

Fig. 5. Original halftone image

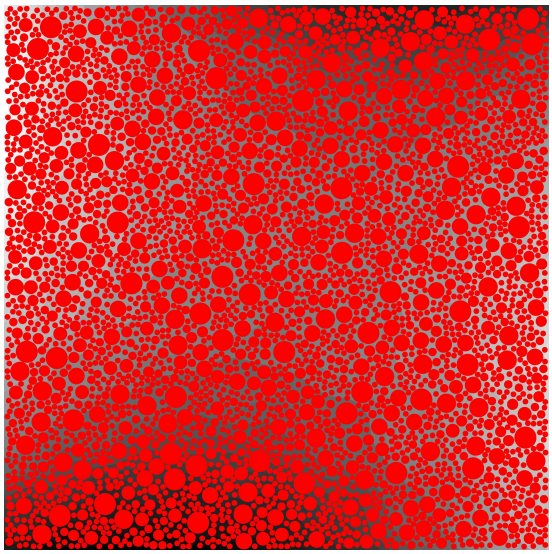

Fig. 6. Corrupted halftone image

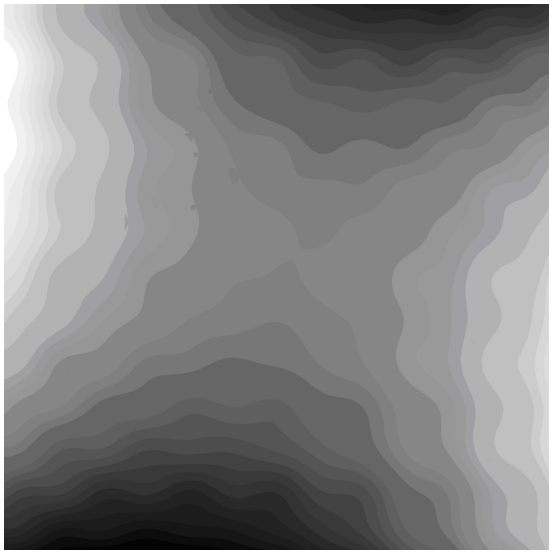

Fig. 7. Restored halftone image

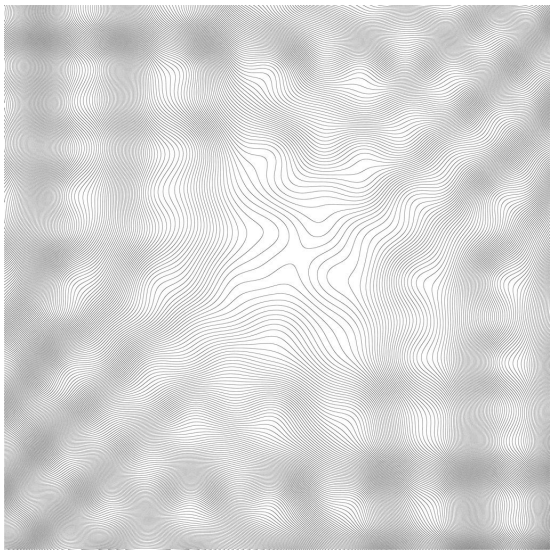

Fig. 8. Original binary image

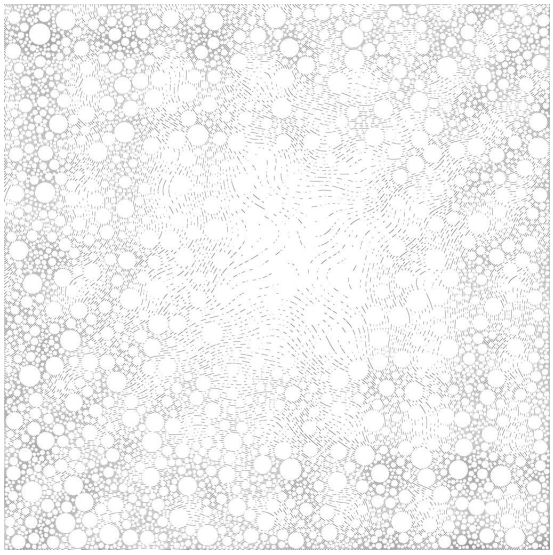

Fig. 9. Corrupted binary image

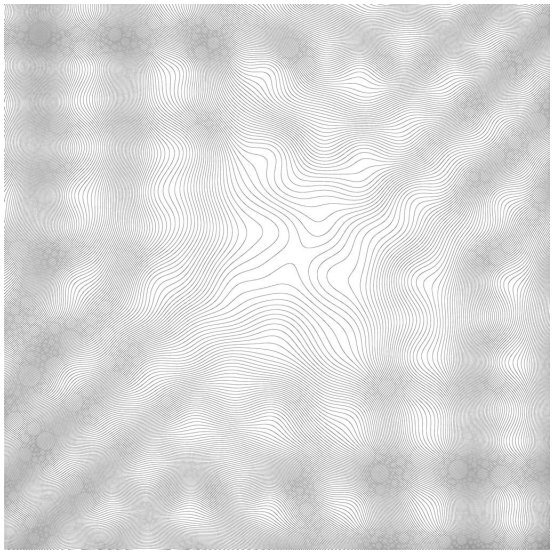

Fig. 10. Restored binary image

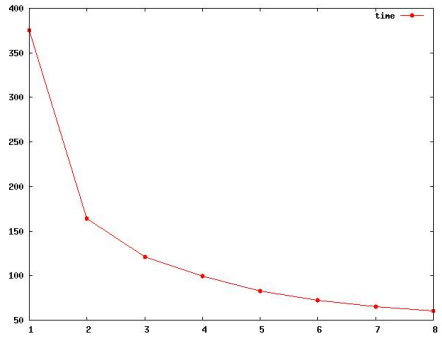

Fig. 11. Plot of time vs number of threads for creation of image

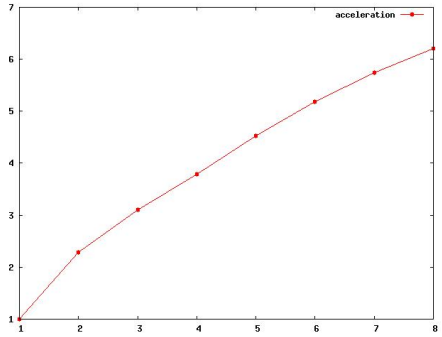

Fig. 12. Plot of acceleration vs number of threads for creation of image

The results of work of `OptimalInpaiting` software prove efficiency of our approach to inpainting (under the assumptions adopted). The authors believe that this approach should be further developed in order to be applied to natural images, and they are looking for possible applications.

The authors thank A.A.Ardentov for participation in the work on this paper.

References

1. The SKIF Supercomputer Project of the Russia-Belarus Union State, `http://skif-grid.botik.ru/`
2. Petitot J.: The neurogeometry of pinwheels as a sub-Riemannian contact structure, J. Physiology - Paris, 97 (2003), 265–309.
3. Petitot J.: Neurogeometrie de la vision — Modeles mathematiques et physiques des architectures fonctionnelles, Editions de l'Ecole Polytechnique (2008)
4. Moiseev I., Sachkov Yu. L.: Maxwell strata in sub-Riemannian problem on the group of motions of a plane, ESAIM: COCV, *ESAIM: COCV*, 16 (2010), 380–399, available at arXiv:0807.4731v1, 29 July 2008.
5. Sachkov Yu. L.: Conjugate and cut time in the sub-Riemannian problem on the group of motions of a plane, ESAIM: COCV, 16 (2010), 1018–1039.
6. Sachkov Yu. L.: Cut locus and optimal synthesis in the sub-Riemannian problem on the group of motions of a plane, ESAIM: COCV, 2011 (accepted).
7. Ardentov A.A., Sachkov Yu.L.: Inpainting via sub-Riemannian minimizers on the rototranslations group, submitted.
8. Duits R., Franken,E.M.: Left-invariant parabolic Evolutions on SE(2) and Contour Enhancement via Invertible Orientation Scores. Part I: Linear Left-invariant Diffusion Equations on SE(2). Quarterly of Applied Mathematics, Volume 68, no. 2, pp. 255-292, June 2008.
9. Agrachev A.A., Boscain U., Gauthier J.P., Rossi F., The intrinsic hypoelliptic Laplacian and its heat kernel on unimodular Lie groups, Journal of Functional Analysis. Vol. 256 (2009). No. 8, pp. 2621–2655.
10. Agrachev A.A., Sachkov Yu. L.: Control Theory from the Geometric Viewpoint, Springer-Verlag, Berlin 2004.