An Interactive Approach to Monocular SLAM

Erick Noe Amezquita Lucio(✉), Jun Liu, and Tony J. Dodd

Department of Automatic Control and Systems Engineering,
University of Sheffield, Mappin Street, Sheffield S1 3JD, UK
{erick.noe,j.liu,t.j.dodd}@sheffield.ac.uk
https://www.sheffield.ac.uk/acse

Abstract. A novel paradigm in SLAM involving user interaction without precise selection using snakes is presented allowing robot and landmark estimation. SLAM and particularly Vision-SLAM rely on automated and algorithm dependent features for robot and landmark estimation. Interactive user input is often not accounted for which could not only provide semantics through object differentiation, giving better map description as well as on-the-fly decision making for a person or robot. General Purpose computing on Graphics Processing Units produce Gradient Vector Flow forces, needed for real-time snake evolution.

Keywords: SLAM · Interactive · GVF · GPGPU

1 Introduction and Related Works

Simultaneous Localisation and Mapping (SLAM) offers many possibilities in robotics, from simple domestic chores to rescue tasks [1]. Current SLAM approaches acquire rigid and algorithm dependent features from the environment using sensors for localisation. However, no effort has been made to incorporate user input. This would allow for on-the-fly semantics, which might differentiate an object with marked importance or state, useful for a person or robot.

Cameras in SLAM have gained momentum due to price and vast image information. Real-time vision-SLAM first used a stereo camera building a sparse 3D map [3]. Simplification resulted in monocular SLAM [4]. Improvements in feature initialisation and non-linearities were obtained using the parallax effect [5]. However, all of these approaches remain automated and dependent on the feature detector used, leading to dense representations with limited map description.

New feature acquisition approaches use curve fitting, mapping places where points are hard to obtain [8], object recognition detects predefined models [6] or offline semantics offering an alternative to dense acquisition [7]. Whilst these show the trend of using simpler approaches in lieu of dense representations, useful semantics and on-the-fly user input is not accounted for, giving information of interest to both user and robot *in the moment* of map creation.

E.N. Amezquita Lucio—This work was supported by CONACYT grant 217509.

C. Dixon and K. Tuyls (Eds.): TAROS 2015, LNAI 9287, pp. 19–25, 2015.
DOI: 10.1007/978-3-319-22416-9_3

Here a novel paradigm in SLAM is introduced: interactive user selection that provides semantics through object differentiation. The key aspects are seen in Figure 1: An user creates an active contour (snake) around a shape of interest (rectangles are used for ease of demonstration). Gradient Vector Flow (GVF) generates forces driving the snake from the image, latching it onto the shape whilst feeding its tracking information to SLAM.

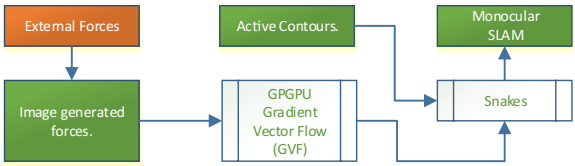


Fig. 1. A novel SLAM approach. A user surrounds a shape creating a snake which later is driven by the object's GVF forces, delivering data to SLAM.

To the best of our knowledge, this is a first interactive approach in SLAM:

- Feature selection is based on the person in lieu of an automated algorithm.
- User semantics in an object might indicate where it comes from.
- On-the-fly semantics differentiating a feature, offering decision making in the moment for a person or robot.
- A first implementation of GVF and snakes into the SLAM methodology.

The interactive methodology is detailed in Section 2 and Section 3 shows results and concludes the paper.

2 Methodology

The overall idea is to make use of an interface, in which an user sets an snake, which tracks a chosen object using attractive forces providing data to SLAM. In parallel General Purpose computing on Graphics Processing Units (GPGPU) produce GVF forces driving the snake. This approach is seen in Algorithm 1.

2.1 Active Contours, Feature Selection

A snake is a contour deformed by external forces. The initial position is set by curve fitting user coordinates, yielding a B-Spline $P(t)$ of given order k [10]:

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t), \qquad t_{\min} \leq t < t_{\max}, \qquad 2 \leq k < n+1, \quad (1)$$

where B_i are n number of control polygon points (see Figure 2); $N_{i,k}$ are recursively obtained basis functions and t is the parameter range approximated through chord lengths [9]. The B-Spline is evolved using [11]:

$$\partial C(p,t)/\partial t = \alpha(p,t)\boldsymbol{T}(p,t) + \beta(p,t)\boldsymbol{N}(p,t), \quad (2)$$

```
Acquire new image;
Interactive user feature selection through user clicks;
Initialise active contour /* Section 2.1 */ ;
for i ← 0 to N do
    Gradient vector flow iteration /* from current image. Section 2.2 */
end
while User input do
    /* user decides when the snake is fully attached to the feature */
    Snake evolution /* Section 2.1 */
end
while SLAM set to start do
    Acquire new image;
    for i ← 0 to N do
        Gradient vector flow iteration /* from current image. Section 2.2*/
    end
    /* active contour follows the feature. Section 2.1 */
    Snake evolution;
    if High-level feature not initialised in SLAM then
        Add feature to state and covariance /* Section 2.3 */
    else
        Obtain observations and update EKF /* Section 2.3 */
    end
end
```

Algorithm 1. Interactive SLAM methodology pseudo code

where $C(p,t)$ is the snake evolution, B_i displacements $\partial C(p,t)/\partial t$ are given by forces affecting $\alpha(p,t)$ and $\beta(p,t)$, $\boldsymbol{T}(p,t)$ is the tangential component of the curve and $\boldsymbol{N}(p,t)$ its inward normal, obtained from the derivatives of (1).

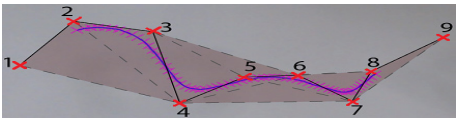


Fig. 2. B-Spline. A cubic ($k = 4$) B-Spline (blue) and its points (purple crosses) lie in the convex hull (segmented line) created by the 9 control polygon points (red crosses). For $k = 2$ the convex hull disappears and the snake can only follow the black line.

2.2 Image Forces and Gradient Vector Flow

GVF forces are created independently in every pixel allowing the snake to become attached onto an object as in Figure 3. This is costly for CPUs but GPUs excel in parallel processing. A GPGPU implementation allows enough recursions at video speed, considering GVF as a function of time [12,13]:

$$u_t(x,y,t_k) = \mu\nabla^2 u_t(x,y,t_{k-1}) - [u_t(x,y,t_{k-1}) - f_x(x,y)] \times \left[f_x(x,y)^2 + f_y(x,y)^2\right],$$
$$v_t(x,y,t_k) = \mu\nabla^2 v_t(x,y,t_{k-1}) - [v_t(x,y,t_{k-1}) - f_y(x,y)] \times \left[f_x(x,y)^2 + f_y(x,y)^2\right],$$

where $u_t(x,y,t_k)$ and $v_t(x,y,t_k)$ are horizontal and vertical forces, with x and y pixel coordinates, these directly affect $\alpha(p,t)$ and $\beta(p,t)$ in (2); μ is a positive value set according image noise, $f_x(x,y)$ $f_y(x,y)$ are obtained from

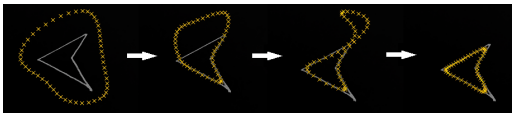


Fig. 3. Active contour evolution: the yellow crosses are points of the snake (sometimes refered as snaxels). Note how it mimics the silhouette of the shape attracts it.

the input binary image's finite differences [13]. Finally $\nabla^2 u_t(x, y, t_{k-1})$ and $\nabla^2 v_t(x, y, t_{k-1})$ are Laplacians applied to previous GVF iterations $u_t(x, y, t_{k-1})$ and $v_t(x, y, t_{k-1})$.

2.3 Monocular SLAM

Once a snake follows a shape its information is used in lieu of algorithm dependent points, often seen in vision-SLAM. Inverse depth Monocular SLAM is used here which uses a camera motion model using constant velocity [5]:

$$\begin{aligned}\mathbf{x}_v &= \left[\mathbf{r}^{wc}_{k+1}, \mathbf{q}^{wc}_{k+1}, \mathbf{v}^{w}_{k+1}, \omega^{c}_{k+1}\right] \\ &= \left[\mathbf{r}^{wc}_{k} + (\mathbf{v}^{w}_{k} + \mathbf{V}^{w})\Delta t, \mathbf{q}^{wc}_{k} \times \mathbf{q}((\omega^{c}_{k} + \Omega^{c})\Delta t), \mathbf{v}^{w}_{k} + \mathbf{V}^{w}, \omega^{c}_{k} + \Omega^{c}\right],\end{aligned}$$

where $\mathbf{r}^{wc}$, $\mathbf{q}^{wc}$ and $\mathbf{v}^{w}$ are respectively the camera position, the quaternion and the linear velocity w.r.t the world frame and ω^{c} is angular velocity w.r.t the camera frame; $\mathbf{q}((\omega^{c}_{k} + \Omega^{c})\Delta t)$ is a quaternion defined by $(\omega^{c}_{k} + \Omega^{c})\Delta t$. Finally $\mathbf{V}^{w}$ and Ω^{c} are zero mean and Gaussian linear and angular velocity impulses $\mathbf{a}^{w}\Delta t$ and $\alpha^{c}\Delta t$, produced by linear and angular accelerations $\mathbf{a}^{w}$ and α^{c} w.r.t the world and camera frame.

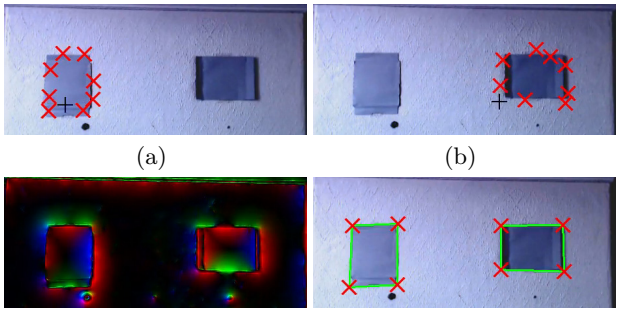
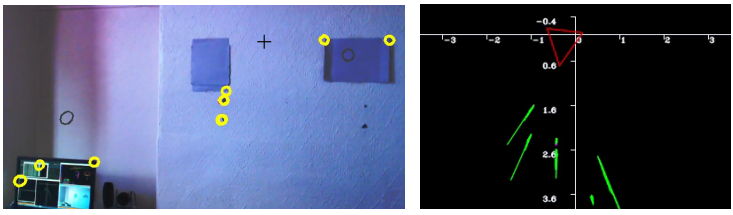


Fig. 4. Feature selection and snake evolution. Figure 4(a) shows a chosen object with red crosses indicating user clicks. Figure 4(b) shows a 2nd feature selected. GVF Forces (washed colours) are seen in Figure 4(c). Figure 4(d) depicts the snakes (green lines) with their control polygon points (red crosses) attracted to GVF's local minima.

Rectangular shapes are used for a simple demonstration of the idea. A snake is set of order $k = 2$ and 4 control polygon points, which fall in the objects' corners (recall Figure 2). Their coordinates are used first as new features $\mathbf{y}_{f_i}$ and then as observations $\mathbf{z}_{f_i}$. The full state vector contains camera and feature states, i.e. $\mathbf{x} = [\mathbf{x}_v \quad \mathbf{y}_{f_1} \quad \mathbf{y}_{f_2} \quad \dots \quad \mathbf{y}_{f_i}]^\top$.



(a) (b)

Fig. 5. Inverse depth monocular-SLAM. Figure 5(a) shows sparse features (yellow circles) from the environment. Map feature representation (green) and camera position (red triangle) is seen in Figure 5(b). Elongated ellipses show big depth uncertainty.

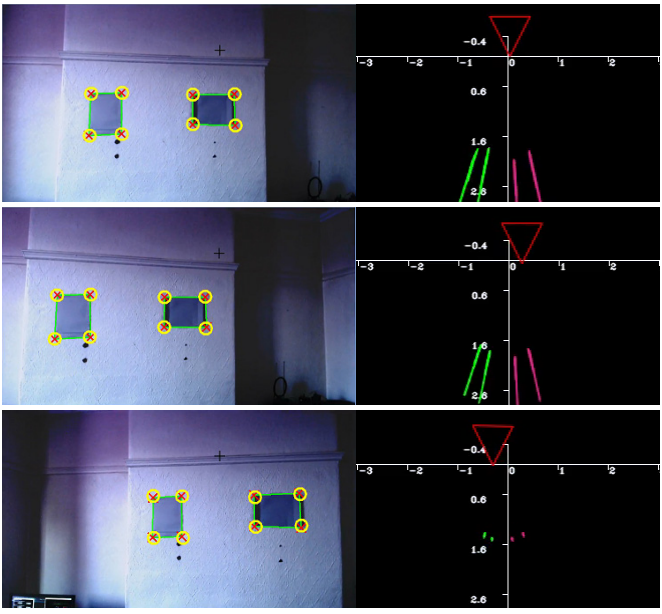


Fig. 6. Camera motion. The left hand images show control polygon points (red crosses) of the snake (green contour), with position uncertainty (yellow). Right hand pictures depict camera (red triangle) and selected objects (green and pink) estimated positions in X and Z planes. Colours allow for semantics through colour differentiation without dense representations or complex recognition.

3 Results

The algorithm was tested on a PC with an AMD FX8350 CPU, AMD R7950 GPU and a Logitech C920 camera. A user is able to judge and surround without precision 2 objects in a wall, about 2 metres from the camera as seen in Figures 4(a) and 4(b). The interface also shows GPGPU GVF forces generated at video rate, with $N = 300$ over an image with 800 by 448 pixels, Figure 4(c). Commands start the initial snakes tracking the objects which are seen as green lines with red crosses for control polygon points in Figure 4(d). This is compared to a baseline inverse depth monocular SLAM seen in Figure 5.

The snakes keep track with camera movements on the left hand side of Figure 6, decreasing uncertainty between camera and objects. The generated map is seen on the right hand side, displaying on-the-fly semantics with different feature colours (green and pink). Thus a novel SLAM paradigm has been presented allowing interactive user input, using active contours and gradient vector flow. This provides semantics through object differentiation (using colours), which can help both robot an user in real-time decision making, whilst avoiding ambiguity of dense mapping. Performance is comparable to a baseline inverse depth monocular-SLAM implementation. Future research will focus on further user on-the-fly interaction in SLAM.

References

1. Durrant-Whyte, H., Bailey, T.: Simultaneous Localization and Mapping: Part I. IEEE Robotics & Automation Magazine **13**(2), 99–110 (2006)
2. Munguia, R., Grau, A.: Camera localization and mapping using delayed feature initialization and inverse depth parametrization. In: IEEE Conf. Emerging Technologies and Factory Automation, Patras, Greece, pp. 981–988, September 2007
3. Davison, A.J., Murray, D.W.: 3D Simultaneous Localisation and Map-Building Using Active Vision. IEEE Trans. on Pattern Analysis and Machine Intelligence **24**(7), 865–880 (2002)
4. Davison, A.J., Reid, I.D., Molton, N.D.: MonoSLAM: Real-Time Single Camera SLAM. IEEE Trans. on Pattern Analysis and Machine Intelligence **29**(6), 1052–1067 (2007)
5. Montiel, J.M.M., Civera, J., Davison, A.J.: Inverse depth parametrization for monocular SLAM. IEEE Trans. on Robotics **24**(5), 932–945 (2008)
6. Salas-Moreno, R.F., Newcombe, R.A., Strasdat, H., Kelly, P.H.J., Davison, A.J.: SLAM++: simultaneous localisation and mapping at the level of objects. In: Proc. IEEE Computer Vision and Pattern Recognition, Portland, OR, pp. 1352–1359, June 2013
7. Siddiqui, J.R., Khatibi, S.: Semantic indoor maps. In: 28th Int. Conf. of Image and Vision Computing, Wellington, New Zealand, pp. 465–470, November 2013
8. Rao, D., Chung, S.-J., Hutchinson, S.: CurveSLAM: An approach for vision-based navigation without point features. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Vilamoura, Portugal, pp 4198–4204, October 2012
9. Rogers, D.F.: An introduction to NURBS with historical perspective. Morgan Kaufmann Publishers, San Francisco (2001)

10. Cox, M.: The numerical evaluation of B-splines. IMA Journal of Applied Mathematics **10**(2), 134–149 (1972)
11. Srikrishnan, V., Chaudhuri, S.: Stabilization of parametric active contours using a tangential redistribution term. IEEE Trans. on Image Processing **18**(8), 1859–1872 (2009)
12. Smistad, E., Elster, A.C., Lindseth, F.: Real-time gradient vector flow on GPUs using OpenCL. Journal of Real-Time Image Processing, 1–8 (2012). ISSN 1861–8200, Online ISSN 1861–8219
13. Xu, C., Prince, J.L.: Snakes, shapes, and gradient vector flow. IEEE Trans. on Image Processing **7**(3), 359–369 (1998)