2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Weighted Node Mapping and Localisation on a Pixel Processor Array

Hector Castillo-Elizalde[1], Yanan Liu[1,2], Laurie Bose[2] and Walterio Mayol-Cuevas[2]

***Abstract*— This paper implements and demonstrates visual route mapping and localisation upon a Pixel Processor Array (PPA). The PPA sensor comprises of an array of Processing Elements (PEs), each of which can capture and process visual information directly. This provides significant parallel processing power allowing novel ways in which information can be processed *on-sensor*. Our method predicts the correct node within a topological map generated from an image sequence by measuring image similarities, spatial coherence, and exploiting the parallel nature of the PPA. Our implementation runs at +300Hz on large public datasets with +2K locations requiring 2.5W at 500 GOPS/W. We compare vs traditionally implemented methods demonstrating better F-1 performance even on simulation. As far as we are aware, we present the first on-sensor mapping and localisation system running entirely on-sensor.**

I. INTRODUCTION

Embedding visual competences reduces latency and opens many agile and interactive applications for robotic platforms. But embedding could reduce generality and hence the favoured approach to date is to use conventional and separate components such as cameras, processors and dataset storage. This paper considers how combining all three into a general, yet close to the signal, Pixel Processor Array (PPA) can deliver the key tasks such as visual image mapping and localisation.

A PPA has a processor for every pixel and allows for massively parallel in-plane computation. This offers novel ways to decompose visual tasks with some parts done at the sensing point itself. Such a notion of in-eye processing exists in many examples in the natural world where eyes do computation (e.g. [1]). PPAs can extend this further as they are re-programmable on the fly and generic enough to implement competences such as visual odometry [2], [3], object tracking [4], agile navigation [5], and CNN classification [6], [7], [8], [9].

Low power localisation and mapping for embedded devices is crucial for many applications. However, due to their architecture, some known approaches like full SLAM [10] call for a different implementation on PPAs vs standard devices. The main advantage of PPAs lies in the ability to efficiently process information at point of capture and in parallel. Approaches like [11], [12], [13] aim to characterise a place based not only on a single image, but a sequence of images. Hence, temporal tracking is often used. Intuitively, an easier prediction of the current position can be done by comparing the recent captured images with the map's information. This implies a linear comparison with n images in the database and leads to a time-consuming process when using a sequential processing method. But with parallel computation and SIMD mechanisms on PPAs, this task is better scaled without costly data bottlenecks.

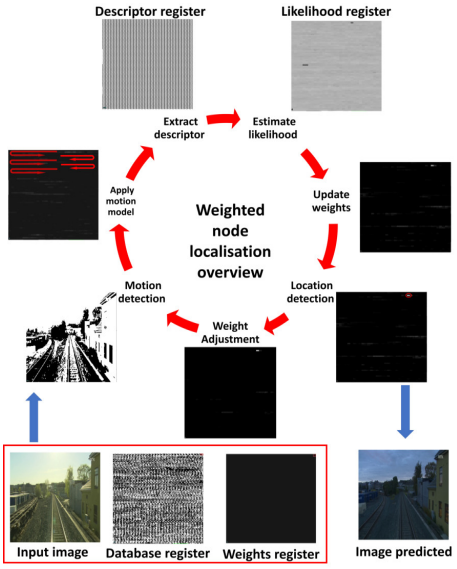


Fig. 1. Overview of the iterative weighted-node process using binary descriptor method.

This paper describes an iterative localisation method on a PPA based on a 1-D topological map. It builds from earlier works for image based mapping but extends and modifies due to the opportunities on PPAs. Our method, while implemented on one PPA, the *SCAMP-5d* [14], is extensible to the general family of sensor-processor devices. We compare the performance of two highly parallelisable methods, low-resolution images and binary descriptors, as prototypical place representations. A topological map is generated from a sequence of images, taking the form of a sequence of connected nodes, each encoding a specific location either as a binary descriptor or using a low-resolution image depending on the method used. This map is stored upon the PPA in the form of a database, using equally sized blocks of processing elements to store each node. The database in register allows comparison with the input image in a fully parallel way. Furthermore, resulting information can be directly transferred within the registers from one node to another, maintaining previous state information. The main contributions of this work are: 1. First implementation of image-centric mapping and localisation fully on the focal plane; 2. A novel weighted

Hector Castillo-Elizalde thanks the Mexican Council of Science and Technology (CONACyT) for sponsoring his master's studies. Part of this work was supported by EPSRC.

[1]Bristol Robotics Laboratory, University of Bristol, BS34 8QZ, UK hectoraaroncastillo@gmail.com

[2]Visual Information Laboratory, University of Bristol, BS1 5DD, UK yanan.liu@bristol.ac.uk

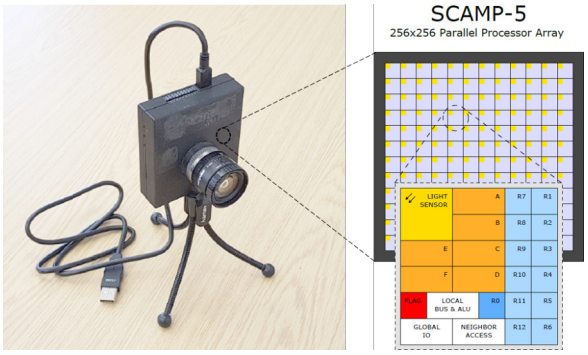


Fig. 2. *SCAMP-5d* vision system and the *SCAMP-5* vision chip architecture. The AREG are shown in orange and marked as A-F while the DREG, R0-R12, are shown in blue. Figure reproduced from [14].

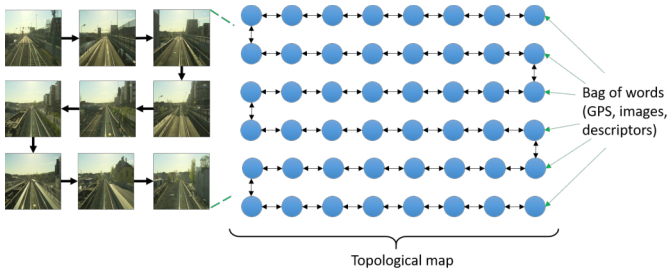


Fig. 3. Topological map built from a bag of words containing different image features. Each blue node represents a place in the environment. As observed, the nodes are concatenated sequentially. In this way, the movement coherence of the image sequence is preserved.

node mapping and localisation algorithm is proposed for parallel implementation; 3. We show favourable comparison with related methods across 3 datasets both in simulation and real PPA hardware.

II. Background and Related Work

A. Pixel Processor Arrays and SCAMP-5d Vision System

A Pixel Processor Array (PPA) has the ability to not only capture visual information at every pixel, but also perform computation on this information "in-pixel". One of the main advantages of a PPA is to directly output useful information extracted from the input image without having to record it. [15] presented a PPA with 256×256 Processing Elements (*PEs*) which can operate on the images at up to 100,000 fps and it is the main component of the *SCAMP-5d* vision system [14]. Besides the PPA, this system includes a dual-core ARM controller to orchestrate, and peripherals described below. The *SCAMP-5d* vision system [14] shown in Fig. 2, consumes approximately 1.5 to 2.5 Watts at power efficiency of 500 GOPS/W. Each *PEs* processors contain 7 analogue registers (AREG), 13 1-bit digital registers (DREG) and an Arithmetic Logic Unit (ALU), which can be seen from architecture of the *SCAMP-5d* vision chip (Figure 2). Operations such as addition, subtraction and division can be performed upon AREG content, while boolean operators such as OR, NOR, NOT or XOR can be performed upon DREG. Additionally, every PE is connected to its four neighbours allowing the transfer of data between them. All PPA instructions are executed using a Single Instruction Multiple Data (SIMD) scheme in which all the PEs execute the same instruction simultaneously in parallel upon their local data. The large number of PEs in *SCAMP-5d* provides significant parallel computational power, with the challenge then being how to effectively harness this capability given the restrictions of SIMD execution.

B. Topological visual localisation

Earlier works on conventional architectures have shown that small amounts of information suffice for place localisation even under substantive changes [13], [11]. Of particular inspiration in these works is to characterise a place as a sequence of images, or sub-route instead of a single image. This allows to recognise a previously seen place using histograms or low-resolution images e.g. 32 pixels and 4 bits depth.

SeqSLAM [12] is a topological version of a SLAM algorithm that uses the idea of sub-routes and low resolution images. The process involves comparing an input image with a database which contains information of previously seen places. After k frames, the most likely place is estimated based on that measurement of similarity. Other works have followed a similar approach, [16] for instance, used binary descriptors rather than low resolution images to characterise a place. Odometry information was integrated with *SeqSLAM* in [17]. Additionally, some other authors [18], [19], [20] used Neural Networks for this task.

Another well-known framework for localisation tasks is factored sampling or particle filters [21], [22]. For the common case of "visual paths", [23] proposed a method which implements a 1-D particle filter and uses the spatial coherence of the images to find the location in a topological map. The idea combines image measurement likelihoods and a motion model. The approach is iterative and consists on estimating the probability density function at each time step. Here, each particle corresponds to a key location in the topological map. The motion is represented with a probability of the robot moving forward one step, staying in the same place or moving more than one place forward; whilst the weights are updated based on a matching likelihood measurement. The result is a score associated to nodes in the map. The idea was extended to 2-D introducing a sequence length in the algorithm presented by [24].

III. A 1-D Weighted Node-Map Algorithm for Visual Route Recognition

Our approach makes use of a topological map (Fig. 3) created from a sequence of images. This map consists of N nodes corresponding to N labelled places. We evaluate two approaches: low-resolution images (8×8 pixels) and binary descriptors (8×4 pixels) for place representation. Each node is thus characterised by either an image or a descriptor, stored as a "database" upon the focal plane of the PPA (such as shown in Fig.6). Furthermore, those nodes have spatial coherence and GPS coordinates could be associated to each of them.

This paper aims to recursively find the node in the topological map closest to the sensor's true location, combining measurements of image similarity and spatial coherence. This is equivalent to the method [23] that iteratively finds the probability of a vehicle to be in a position given a measurement of similarity and a motion model, $p(x_t|z_t, U^t)$. In our case, considering the topological map, $\boldsymbol{X}_t$, a set of nodes $x_t^{[t]}$ at the time t,

$$\boldsymbol{X}_t = x_t^{[1]}, x_t^{[2]}, ..., x_t^{[N]} \tag{1}$$

where N is the total number of nodes. Each of them has an associated weight value ${w_t}^{[i]}$, $\boldsymbol{W}_t$ represents the set of weights for the set of nodes $\boldsymbol{X}_t$. Those nodes and weights are represented on the *SCAMP-5d* using blocks of AREG across multiple PEs, as shown in Fig. 4. All weights are updated simultaneously at every time-step based on a likelihood measurement in a parallel manner. The nodes in the topological map are sequentially connected, with each representing a specific place in the space. Thus a "forward" movement from node $n-1$ moves towards the position of node n. Or towards the node $n-2$ for a "backward" movement. When such movement is detected, the weights for each node are similarly moved to the next or previous node to preserve coherence.

Our approach is outlined in Algorithm 1 and is inspired by Liu & Zhang [23], but with significant modification and optimisation due to the unique PPA hardware architecture, as explained in the following sections.

Algorithm 1: Weights estimation algorithm

```
if (database_exists) then
    load_database();
else
    mapping();
end
t ← 0;
initialise X_t;
while input_frame do
    if (motion_model()) then
        for all the weights i = 1, ..., N ;
        x_t^[i] = motion_model(x_{t-1}^[i]) ;
        μ_t^[i] = likelihood(z_t|x_t^[i]);
        X_t = X_{t-1} + (x_t, μ_t);
        if (t => sequence_length) then
            prediction = max_weight(X_t) ;
        end
        if (t % sequence_length == 0) then
            w_t^[i] = w_t^[i] − γmax(W_t);
            X_t = X_t + (W_t);
        end
        t ← t + 1;
    end
end
```

A. Initialisation

In this first step, both database and weights are initialised within the PPA's registers, with the database consisting of node information about the previously seen places. Effectively, the PE array of the PPA is divided up into equally sized blocks of PEs, each of them storing information about a specific node in AREG. The size of each block depends on the processed image dimension or the descriptor length. In this work, two sizes of node are used, 8×8 and 4×8 for low-resolution image method and binary descriptor method respectively, Fig. 4 shows this concept.

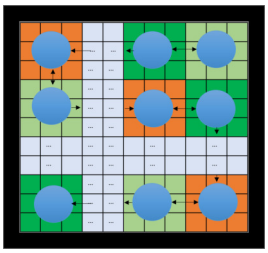


Fig. 4. Nodes in the AREG stored sequentially showed in different colours. As mentioned, each node is a collection of units. Each of the units represent the node weight in a range value of [-128,127].

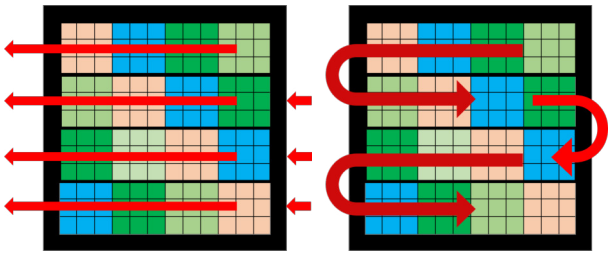


Fig. 5. Shifting process in the weighted nodes. (a) shows the first option to perform the shifting, where all the weights are sorted in the same way. (b) shows the *snake* way to store and shift the weights.

B. Motion model estimation

Our approach combines two different hypotheses for localisation: image similarities and a motion model. Here, we considered the motion model in a similar way as in [23] but simplified to two possible options: moving forward or stay in the same position. Moreover, due to the high frame rate in the *SCAMP-5d*, the idea of having a probability value associated to the movement is replaced by an estimation based on the pixels in two consecutive frames. Assuming the *SCAMP-5d* is pointing forward, then a forward movement can be identified by examining the motion of elements away from the centre of the image. A series of binary operations between the current image and previous ones can give a quick if rough estimate of such motion. This 1-D approach uses the motion model in the following way: moving forward is associated with the belief of each weight travelling to the next node in the topological map. The nodes and their weights are stored sequentially in the PE array starting from the rightmost upper corner, hence moving forward means shifting to the left and down. We borrow these ideas from [9] to store the blocks which are sorted in a way known as a *snake* stack such that the first weight in the next row is just below the last place of the previous row as in Fig. 5. This *snake* stack shifting is not only faster but prevents the information loss in the AREG.

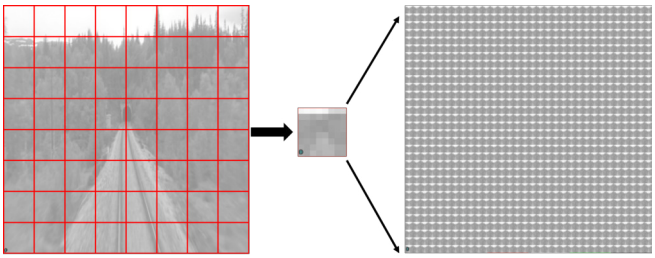


Fig. 6. Resolution reduction in the greyscale image. The average values in the input image are calculated using a grid of 8×8 shown in red on the left image and then use the mean values as pixel values (centre image). This image is then replicated to a whole register to generate image differences in parallel.

C. Temporal weight value estimation

At this stage, the motion model has been calculated and applied to the set of nodes $\boldsymbol{X}_t$. The next step is to estimate the temporal weights which are related to the likelihood of the current input measurement (either low-resolution image or binary descriptor method), z_t, being at that node in the topological map. In this work, two different image pre-processing methods are used: Sum of Absolute Differences (SAD) of low-resolution images with 64 pixels images and binary descriptors of 32 bits.

The method to calculate the SAD between the current input image and all those in the database is similar to the approach of [11]. Taking advantage of the PPA hardware architecture, these comparison operations can all be performed in parallel. Hence, the data should be placed in a specific way: consider the case where images have the dimension of 8×8 pixels; if this image is placed into an AREG, it will occupy 64 pixels within an 8×8 block. Given the *SCAMP-5d* resolution, 256×256, it means a total of 32×32 images of 8×8 which could be processed simultaneously in that AREG. This implies that with this size of images, the maximum number of nodes N is 1024 on one AREG. Here, it is essential to say that the database must be loaded in the same way the images are stored, using the *snake* stack pattern. At every iteration, when a new greyscale image is captured, it is down-sampled to 8×8. This is done by simple average downsampling, as shown in Fig. 6.

The database is created using the *snake* pattern in a different AREG. Therefore, the input image could be compared with the entire database by performing a subtraction followed by an absolute operation between database and input image AREG (both downsampled). This operation is defined by equation 2.

$$likelihood(z_t|x_t^{[i]}) = SAD = \sum_{i=1}^{8}\sum_{j=1}^{8}|I_{input} - I_{db}| \quad (2)$$

where I_{input} is the input image; I_{db} represents all images in the database; and i and j represent the rows and columns in each block. A SAD is obtained for each block, which will be related to each node's weight. In the binary descriptor version of our algorithm the database is formed of a variation of the *LBP* descriptor [25]. This descriptor is calculated within a block of 6×6 pixels and the resulting binary string is used to characterise the whole patch. Robustness is introduced by multiple gridding [26] which splits the initial block into smaller units. Thus, the original 6×6 block is divided into four 3×3 blocks for which the *LBP* descriptor is calculated. This leads to 4 concatenated descriptors of 8 bits long per 6×6 block. The descriptor for each node of the map is then stored in a DREG within that node's block of PEs. The likelihood for each node can be then computed in parallel, calculating the Hamming distance between database and the input image descriptor registers.

D. Updating node weights

Line 13 in algorithm 1 states that the node weights values are updated based on the motion model and the temporal weight estimation. The node weight shifting applies the motion model described earlier, and the temporal weights at this point have been already calculated. Here it is essential to introduce a new parameter, α, which benefits the estimation of the new weights values. For low-resolution images, the likelihood is calculated by the SAD. However, image similarity can be represented with Hamming distance for binary descriptors. In both cases, the smaller the value, the more similarity between two images. Equation 3 transform the temporal weight values to a better representation for *SCAMP-5d*.

$$w_t^{[i]} = w_{t-1}^{[i]} + (\alpha - \mu_t^{[i]}) \quad (3)$$

A likelihood of 0 indicates identical images, then $w_t^{[i]}$ is equal to the previous weight plus α. On the other hand, as the likelihood increases, the value of the weight decreases.

E. Location detection

The result of estimation needs to be verified after each iteration, which means all the weights need to be scanned to find the one with the highest value associated with the correct position. [11] proposed the sequence-based visual localisation method, which means that a position can be estimated with a sequence of images (known as subroute) rather than relying on a single image. Considering the 8-bit AREG in the PPA has a value range of [-128,127], and the weights are accumulative, it subjects to saturating after several additions resulting in more than one weight with the highest value. Here, a condition of continuity is defined: the predicted node is compared to the result in the $t-1$ time step; the node with the smallest Euclidean distance to the previous belief is chosen as correct.

F. Weight Adjustment

To prevent saturation of the AREG storing the weights, we adjust them after every sequence of k images as follows. The highest weight value is multiplied by a factor γ, which is within $[0-1]$; this value is then subtracted from the weights. A new set of node weights is created by applying equations 4 and 5.

$$w_t^{[i]} = w_t^{[i]} - \gamma max(\boldsymbol{W}_t) \quad (4)$$

$$\boldsymbol{X}_t = \boldsymbol{X}_t + (\boldsymbol{W}_t) \quad (5)$$

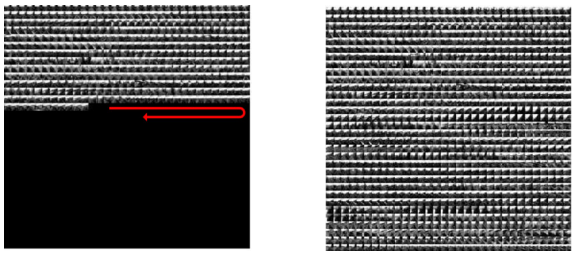


Fig. 7. Mapping procedure. The image in the left shows the procedure where a place on the space is characterised by a low-resolution image; this image is stored in a sequential manner following the *snake* pattern. The image on the right shows the full node map or also named database.

In summary, Fig. 1 shows the iterative process using binary descriptors. It starts from loading the database, the input image and initialise the weights register. For binary descriptor method, each weight is represented by a block in size of 4×8 PEs. Then, the motion detection function gives the motion model based on the input image, which is applied in the next step. Afterwards, the descriptor of the current input image is extracted and replicated through all the register. The following step computes the likelihood by getting the Hamming distance between the image descriptor and the database register. Then, the values of the weights are updated. Later, the location detection finds the node weight with the highest value, the brightest one, and gives a prediction of the image from the database, followed by the weight adjustment step.

G. Mapping

Mapping in our approach takes the form of database creation from a sequence of input images. Depending on the method used to represent locations in space, the mapping process will extract information, either 8×8 images for the low-resolution image method, or 32 bit binary descriptors. In both cases the information for each image from the sequence is placed in the corresponding node, following the *snake* stack pattern (Fig. 7).

IV. Experimental Results

This section shows the results of our implementation previously presented. Localisation tasks are shown across three datasets, two of which are from outdoor environments and one from a simulated laboratory on a mobile robot. Table I shows a comparison of F1 scores across the different implementations and datasets. This measurement is used to combine precision and recall information into one value which describes the accuracy of the system.

A. Nordland dataset [27]

This dataset records train rides in Norway of length 728 km [28]. With recordings over four different seasons as shown in Fig. 9. In this work, we perform tests both on the *SCAMP-5d* simulator [29] and the real *SCAMP-5d* hardware for two seasons, fall and summer, and compare our results with other methods in terms of accuracy and efficiency (Table I). The *SCAMP-5d* simulator is used to validate the performance of our proposed algorithms without the hardware-dependant errors currently encountered on the experimental SCAMP-5d

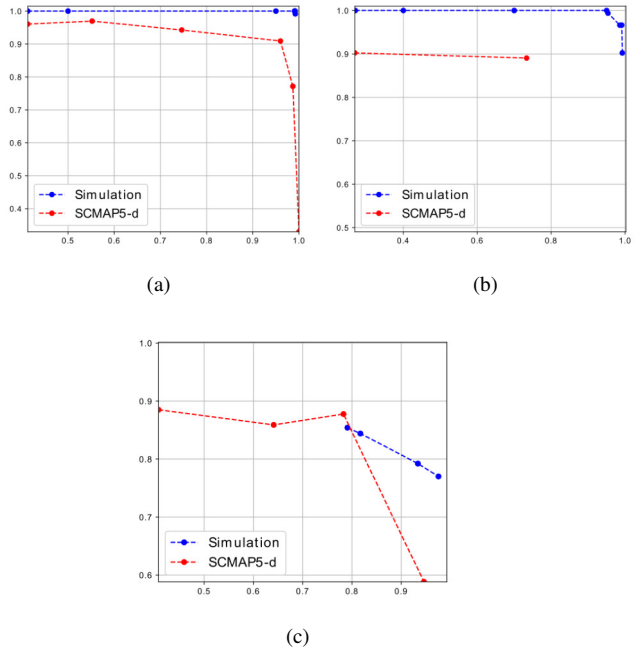


Fig. 8. Precision-Recall result for the weighted node approach in simulation (*SCAMP-5d* simulator) and *SCAMP-5d* with the different datasets. (a) performance using the binary descriptor method with the Nordland database. (b) performance with Oxford dataset using low-resolution image method. (c) performance in the simulated lab environment with the binary descriptor method.

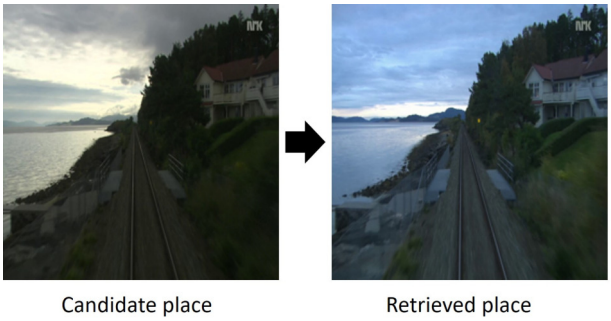
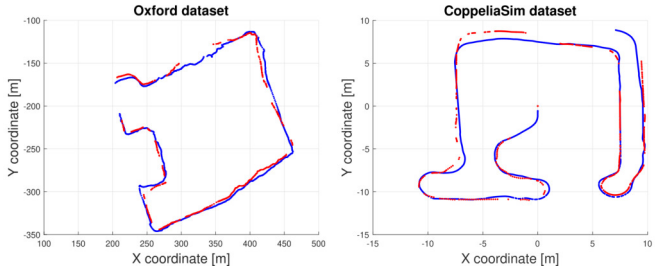


Fig. 9. A recognised place by our weighted node approach in the Nordland dataset. As observed, the input place is an image taken during the summer season, while the result is the estimation of an image in the fall season.

hardware. Fig. 8a illustrates the performance gap between these implementations. The simulation is close to achieve the maximum level of recall at maximum level of precision, which validates our algorithm. Both approaches maintain a value of precision higher than 0.9 at a level of 0.9 of recall. The F1-score is presented in Table I. Results indicate better performance on simulation compared with competing and traditionally implemented methods of *ABLE-M*[16], *SeqSLAM* [28] and Liu & Zhang [23]. Figure 9 shows an example of the input location to the algorithm and its prediction. Additional results can be found on the supplementary video from `https://youtu.be/daFa4iFeQYM`.



(a) (b)

Fig. 10. (a) Position estimated by our proposed low-resolution images approach implemented on *SCAMP-5d* with Oxford dataset. (b) Position estimated by binary descriptor approach when implemented in the *SCAMP-5d* testing with images generated by the simulated environment. Blue points represent the locations of the input image and red points the correctly predicted locations.

B. Oxford dataset [30]

Taken by a mobile robot moving through Oxford city centre, this dataset records images from a camera moving from left to right taking a shot at each time step. The Oxford dataset was previously used in [31] and [23], using a particle filter approach to which we can compare. Figure 8b shows the performance of the our approach in this database in both, simulation and real hardware. While the simulation data has a high performance in terms of precision, it is always above 90%, the implementation on the *SCAMP-5d* shows a gap with the simulation as expected, its maximum value is near 90% precision. The image prediction location is shown in Fig. 10a.

C. Simulated lab environment

We created this dataset for greater control of the scene with CoppeliaSim [32], a robot simulation environment, with a *scene* shown in Fig. 11. The idea is first to collect visual information within the scene and then introduce changes to the environment for testing robot relocalisation. Different from our previous tests, in Fig. 8c, we observe a narrower gap between simulation and the *SCAMP-5d* for the binary descriptors approach. Both approaches achieve a precision close to 90%. However, in terms of recall, it is higher in simulation, which leads to a better overall performance in table I. Nevertheless, the simulation F1 score is just 3.43% higher than that in real hardware. Similar to the Oxford dataset, most of the fails in localisation happen when turning or in corners as shown in figure 10b, which result from the high amount of changes when the camera rotates in those sections.

D. Processing time

Our fully parallel algorithm takes 3.17 milliseconds (ms) (315 fps) for the local binary pattern method and 2.59 ms (386 fps) for the low-resolution image method respectively. Note that other methods such as [16], [23] do deploy their systems on powerful external conventional computers. But while the PPA concept and hardware is still in development, its fundamental parallel nature surpasses their performance, for current hardware requires low power consumption ($\leq$ 2.5W) and performs most computations on-sensor.

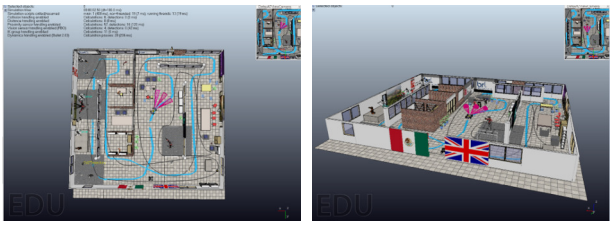


Fig. 11. (a) Top view of the *scene* in CoppeliaSim which simulates a laboratory environment. (b) Front view of the laboratory scene.

TABLE I

F1-SCORES AND LOCALISATION TIME EVALUATION.

Dataset	Baselines	Ours simulation	Ours SCAMP-5d HW
Nordland	*ABLE-M*[16] 97%	Binary descriptor **99.57%** Low-resolution images 98.16%	Binary descriptor **93.38%** Low-resolution images 71.78%
	9.3 ms		
	SeqSLAM [28] 92%		**3.168 ms**
Oxford	Liu & Zhang[23] 93.04%	Binary descriptor **98.98%** Low-resolution images 97.84%	Binary descriptor 64.95% Low-resolution images **80.4%**
	800 ms		**2.59 ms**
CoppeliaSim	——	Binary descriptor 86.13 % Low-resolution images **95.03 %**	Binary descriptor **82.7 %** Low-resolution images 82.6 %
			3.168 ms

V. CONCLUSIONS

In this work we presented a novel parallel node weighting approach for performing visual route mapping and localisation upon a dynamically re-programmable PPA sensor-processor. As far as we are aware, it is the first on-sensor mapping and localisation demonstration. By embedding a dataset into the registers of the sensor's processing array, our approach is able to perform the majority of computation efficiently in parallel upon the focal plane of the sensor itself. Specifically, we demonstrated two map representations: binary descriptors and low-resolution images. Results were demonstrated across public datasets, illustrating how different encoding representations vary in localisation performance. The PPA hardware used is of a prototype nature, and we still observe a performance gap between simulation and real PPA hardware due to implementation constraints. However, we can still demonstrate our node weighted localisation algorithms outperforms baseline approaches like SeqSLAM and ABLE-M, and demonstrates the potential of such on-sensor processing. It is our hope that this work illuminates an upcoming trend of novel visual architectures where perception and computation can be closely integrated efficiently, which opens new opportunities for complex and agile robotic tasks.

REFERENCES

[1] J. Y. Lettvin, H. R. Maturana, W. S. McCulloch, and W. H. Pitts, "What the frog's eye tells the frog's brain," *Proceedings of the IRE*, vol. 47, no. 11, pp. 1940–1951, 1959.

[2] L. Bose, J. Chen, S. J. Carey, P. Dudek, and W. Mayol-Cuevas, "Visual odometry for pixel processor arrays," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4604–4612.

[3] R. Murai, S. Saeedi, and P. H. J. Kelly, "Bit-vo: Visual odometry at 300 fps using binary features from the focal plane," 2020.

[4] C. Greatwood, L. Bose, T. Richardson, W. Mayol-Cuevas, J. Chen, S. J. Carey, and P. Dudek, "Tracking control of a uav with a parallel visual processor," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 4248–4254.

[5] Y. Liu, L. Bose, C. Greatwood, J. Chen, R. Fan, T. Richardson, S. J. Carey, P. Dudek, and W. Mayol-Cuevas, "Agile reactive navigation for a non-holonomic mobile robot using a pixel processor array," *IET image processing*, pp. 1–10, 2021. [Online]. Available: https://doi.org/10.1049/ipr2.12158

[6] L. Bose, P. Dudek, J. Chen, S. Carey, and W. Mayol-Cuevas, "Fully embedding fast convolutional networks on pixel processor arrays," Aug. 2020, 16th European Conference on Computer Vision, ECCV20 ; Conference date: 23-08-2020 Through 28-08-2020.

[7] M. Z. Wong, B. Guillard, R. Murai, S. Saeedi, and P. H. J. Kelly, "Analognet: Convolutional neural network inference on analog focal plane sensor processors," 2020.

[8] Y. Liu, L. Bose, J. Chen, S. Carey, P. Dudek, and W. Mayol-Cuevas, "High-speed light-weight cnn inference via strided convolutions on a pixel processor array," in *BMVC 2020 Programme*. British Machine Vision Association, Jul. 2020.

[9] L. Bose, J. Chen, S. J. Carey, P. Dudek, and W. Mayol-Cuevas, "A camera that cnns: Towards embedded neural networks on pixel processor arrays," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1335–1344.

[10] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[11] M. Milford, "Visual route recognition with a handful of bits," *Proc. 2012 Robotics: Science and Systems VIII*, pp. 297–304, 2012.

[12] M. J. Milford and G. F. Wyeth, "Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 1643–1649.

[13] H. Aoki, B. Schiele, and A. Pentland, "Realtime personal positioning system for wearable computers," in *2012 16th International Symposium on Wearable Computers*. IEEE Computer Society, oct 1999.

[14] J. Chen, S. J. Carey, and P. Dudek, "Scamp5d vision system and development framework," in *Proceedings of the 12th International Conference on Distributed Smart Cameras*, 2018, pp. 1–2.

[15] S. J. Carey, A. Lopich, D. R. Barr, B. Wang, and P. Dudek, "A 100,000 fps vision sensor with embedded 535gops/w 256× 256 simd processor array," in *2013 Symposium on VLSI Circuits*. IEEE, 2013, pp. C182–C183.

[16] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, and E. Romera, "Towards life-long visual localization using an efficient matching of binary sequences from images," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 6328–6335.

[17] Y. Wang, X. Hu, J. Lian, L. Zhang, and X. Kong, "Improved seq slam for real-time place recognition and navigation error correction," in *2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics*, vol. 1. IEEE, 2015, pp. 260–264.

[18] M. Milford, C. Shen, S. Lowry, N. Suenderhauf, S. Shirazi, G. Lin, F. Liu, E. Pepperell, C. Lerma, B. Upcroft *et al.*, "Sequence searching with deep-learnt depth for condition-and viewpoint-invariant route-based place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 18–25.

[19] B. Dongdong, W. Chaoqun, B. Zhang, Y. Xiaodong, Y. Xuejun *et al.*, "Cnn feature boosted seqslam for real-time loop closure detection," *Chinese Journal of Electronics*, vol. 27, no. 3, pp. 488–499, 2018.

[20] M. Milford, H. Kim, M. Mangan, S. Leutenegger, T. Stone, B. Webb, and A. Davison, "Place recognition with event-based cameras and a neural implementation of seqslam," *arXiv preprint arXiv:1505.04548*, 2015.

[21] S. Thrun, "Particle filters in robotics," in *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2002, pp. 511–518.

[22] Y. Liu, R. Fan, B. Yu, M. J. Bocus, M. Liu, H. Ni, J. Fan, and S. Mao, "Mobile robot localisation and navigation using lego nxt and ultrasonic sensor," in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 1088–1093.

[23] Y. Liu and H. Zhang, "Visual loop closure detection with a compact image descriptor," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1051–1056.

[24] Y. Liu and H. Zhang, "Towards improving the efficiency of sequence-based slam," in *2013 IEEE International Conference on Mechatronics and Automation*. IEEE, 2013, pp. 1261–1266.

[25] F. Juefei-Xu and M. Savvides, "Subspace-based discrete transform encoded local binary patterns representations for robust periocular matching on nist's face recognition grand challenge," *IEEE transactions on image processing*, vol. 23, no. 8, pp. 3490–3505, 2014.

[26] X. Yang and K.-T. Cheng, "Ldb: An ultra-fast feature for scalable augmented reality on mobile devices," in *2012 IEEE international symposium on mixed and augmented reality (ISMAR)*. IEEE, 2012, pp. 49–57.

[27] "single-view place recognition under seasonal changes." [Online]. Available: https://webdiis.unizar.es/~jmfacil/pr-nordland/

[28] N. Sünderhauf, P. Neubert, and P. Protzel, "Are we there yet? challenging seqslam on a 3000 km journey across all four seasons," in *Proc. of Workshop on Long-Term Autonomy, IEEE International Conference on Robotics and Automation (ICRA)*, 2013, p. 2013.

[29] J. Chen. (2020) Guide for scamp5d simulation. [Online]. Available: https://personalpages.manchester.ac.uk/staff/jianing.chen/scamp5d_lib_doc_html/_p_a_g_e__s_i_m_u_l_a_t_i_o_n.html

[30] "Multimedia extensions for fab-map: probabilistic localization and mapping in the space of appearance." [Online]. Available: http://www.robots.ox.ac.uk/~mobile/IJRR_2008_Dataset/data.html

[31] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[32] E. Rohmer, S. P. Singh, and M. Freese, "Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework," in *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013.