2018 7th IEEE International Conference on Biomedical
Robotics and Biomechatronics (Biorob)
Enschede, The Netherlands, August 26-29, 2018

An Insect-Sized Robot that uses a Custom-Built Onboard Camera and a Neural Network to Classify and Respond to Visual Input

Sivakumar Balasubramanian[1] Yogesh M. Chukewad[1] Johannes M. James[1] Geoffrey L. Barrows[2]
Sawyer B. Fuller[1]

***Abstract*— To date, controlled flight of very small, insect-sized (∼100 mg) Micro Aerial Vehicles (MAVs) has required off-board sensors and computation. Achieving autonomy in more general environments that do not have such resources available will require integrating these components. In this work we present advances toward this goal by demonstrating a new, custom-built, low-weight (26 mg) camera mounted on a 74 mg flapping-wing robot. We implemented a convolution neural network (CNN) to classify images. Using this system, we demonstrated how the camera-equipped robot could repeatably move toward flower images and away from predator images. An estimate of the computational requirements of our network indicates that it could be performed using low-weight micro-controllers that are compatible with the payload and power constraints of insect-scale MAVs. Many desired capabilities for aerial vehicles, such as landing site selection and obstacle detection and avoidance, are ill-defined because the boundary between positive and negative classes are unclear. This work shows that CNNs operating on input from vision, which have previously been deployed only on larger robots, can be used at insect-scale for such tasks.**

I. INTRODUCTION

Autonomous flight control of insect scale MAVs has thus far required using external motion capture cameras and computation [1]. This limits the flight to be only within the arena. To make the robot compatible with a diverse set of environments, sensors and computation should be brought onboard. As robot size decreases, the low resolution of Global Positioning System (GPS), which is 1–10 m at best, makes it impractical for flight control. Vision sensors provide a better alternative because they do not have these constraints and are low-weight. They have previously been used successfully in GPS-denied environments on rotating-wing aircraft for navigation and guidance [2]–[5], autonomous mapping [6], and feature based stereo visual odometry [7]. So far this has not been achieved at an insect scale in full 3-D due to difficulty in down-scaling all the components. Previous work at insect scale demonstrated an integrated camera, but robot motion was constrained to one degree of freedom along guide wires [1].

Here we are interested in achieving full visual flight control at insect scale, which starts with a characterization of our physical robot. The University of Washington (UW)

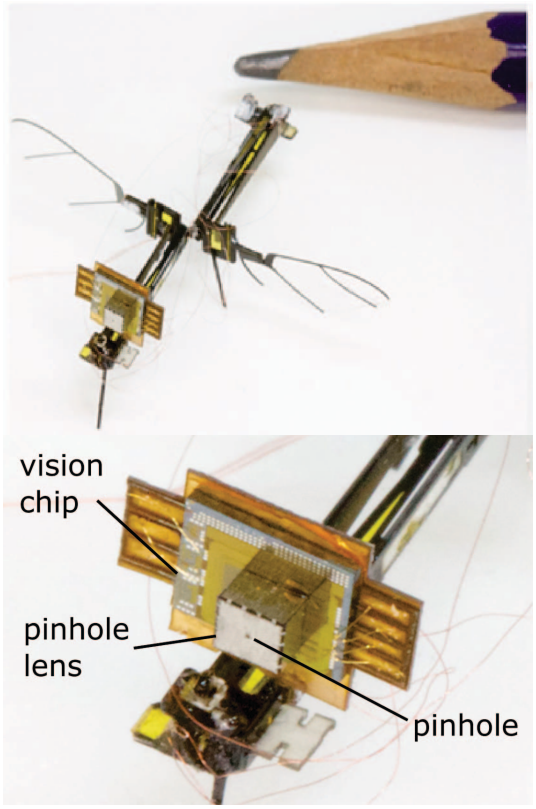

Fig. 1. The insect scale robot (UW RoboFly) with the fully fabricated flight weight camera mounted on it. The Pinhole setup has a focal length of 2 mm and and pinhole diameter of 0.1 mm (top). Close up view of the flight weight camera (bottom).

RoboFly is a 75 mg flapping wing robot shown in Fig. 1 [8]–[10]. It is designed and fabricated at the Autonomous Insect Robotics (AIR) Laboratory at the University of Washington, Seattle. It is fabricated using a Diode Pumped Solid State (DPSS) laser to cut an 80 mm thick carbon fiber composite which is then folded into shape. It uses bimorph piezo actuators for flapping its wings at high frequency (140 Hz) for generating the required lift. The RoboFly can perform aerial as well as ground locomotion by flapping its wings, due to lowered center of mass as compared to earlier versions of the insect robots [11].

Because of its scale, angular acceleration rates of the RoboFly are much higher than for larger drones [12] . For example, a 0.5 kg quadrotor-style helicopter, the Ascending Technologies X-3D, can perform angular accelerations up to approximately 200 rads/s^2 for performing multi-flip maneu-

*The Centeye vision chip was partially supported by the National Science Foundation (award no. CCF-0926148). Any opinions findings or conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

[1]The authors are with the Department of Mechanical Engineering, University of Washington, Seattle, WA 98105 sivabala@uw.edu

[2]Geoffrey L. Barrows is the founder of Centeye Inc.

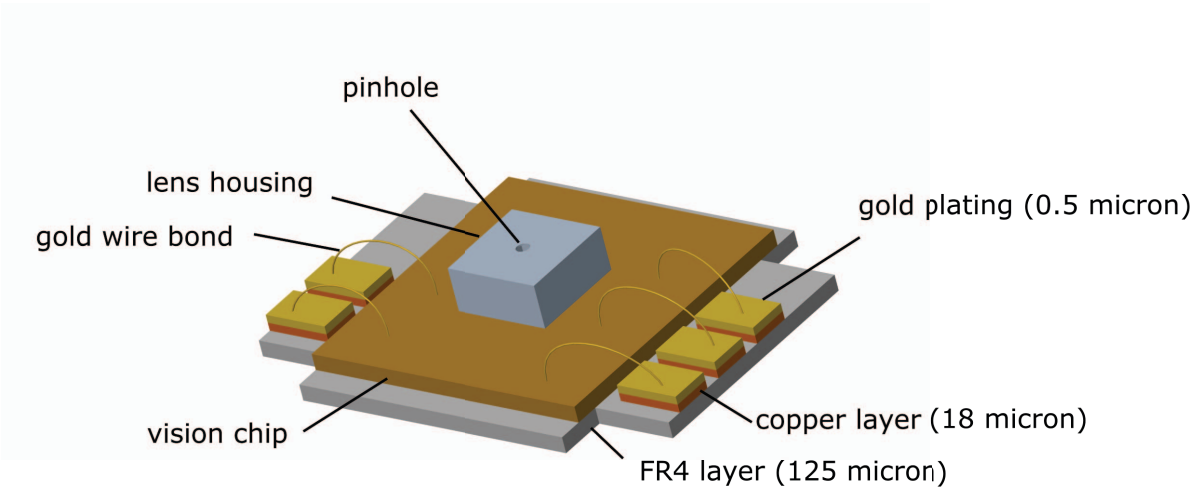

Fig. 2. Diagram of the fully fabricated camera. The vision chip is adhered to an FR4 sheet that has been patterned with isolated pads of gold plated copper. Gold wire bonds connect the vision chips to the pads. A pinhole lens is mounted over the pixel array on the vision chip to focus light rays.

vers [13], while a much smaller quadrotor, the Crazyflie 2.0 at about 30 g can do so at a higher angular acceleration of 500 rads/s^2 [14]. By comparison, UW RoboFly can achieve approximately 1400 rads/s^2 and 900 rads/s^2 around the roll and pitch axes respectively [11]. A light weight visual sensor compatible with these speeds is needed to perform flight control.

We show the design and fabrication of a low-weight camera with a 2-D vision sensor integrated onto the RoboFly. It has a pixel resolution of 64 $\times$ 64 and a weight of 26 mg. It is used to classify images as predators or flowers, and the robot's motion is determined based on the classification feedback. The image classification is performed using a CNN. Neural networks have been shown to match human performance in image recognition [15] and also perform better than other classifiers [16]. Our approach minimizes layers and features of the CNN to reduce computation so that it is compatible with limited on-board computation capability owing to limited battery power and weight constraints of insect scale. Ultimately our goal will be to use this camera for flight control. Here we use our robot's ability to move on the ground as a first step to validate our sensors and algorithms.

In section II, the fabrication and interface of the low weight camera is discussed. Analysis of the CNN classification task performed using the camera is provided in section III. Section IV gives details of an experiment with the camera on board the insect robot.

II. Fabrication and Interface

The camera consists of a bare die vision sensor, the circuits that interface the data from the vision sensor to a development board, and a pinhole lens. The vision sensor is manufactured by Centeye Inc. (Whiteoak model, Washington, DC), and is designed specifically for the use in insect-scale flight control. It consists of a 64 $\times$ 64 pixel array of 12.5 μm sized photo-diodes that capture light intensity values. A 5-wire interface provides power, control/reset commands, and an analog pixel reading. To reduce weight to a minimum, we use a pinhole lens to eliminate the mass of an ordinary lens. Fig. 1 shows the RoboFly with the camera mounted on-board and Fig. 1 shows a closeup view of the camera.

The RoboFly shown in Fig. 1 consists of several structural and functional components such as airframes, transmissions, actuators and wings. In this design, the airframe and transmission are all assembled from a single laminate in order to improve the accuracy and precision of folds. With the help of specially designed folding, this design limits the error to only the rotation about the folding axis. The details about the folding procedure involved are presented in [10].

A. Camera Fabrication

A layout of the camera is shown in Fig. 2. The vision chip is adhered to a flexible printed circuit board made of copper-clad FR4 plated with gold. The printed circuit was fabricated by first electroplating copper-clad FR4 with gold and then ablating these metals using the DPSS laser. We connected the pads on the chip to the substrate using a ball bonder and a 25 μm gold wire. Gold wires are used as they provide

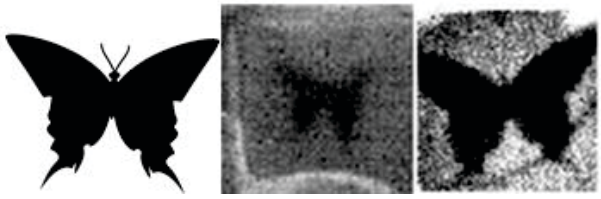

Fig. 3. A black image of a butterfly printed on a white sheet(left), image captured by the flight weight camera with the pinhole setup of 2 mm focal length (middle), image captured by the flight weight camera with the pinhole setup of 4 mm focal length (right). The 2 mm focus pinhole setup has a wider field of view compared to the 4 mm focus pinhole setup

good corrosion resistance and high thermal conductivity. The gold plating provides better bondability for gold wires.

B. Pinhole Setup

The pinhole lens was cut using the DPSS laser from a 50 μm thick stainless steel shim and folded into shape. The inner surface of the pinhole lens was painted black to avoid reflections from the steel surface. After folding the shim to the desired cubical shape, the edges were covered with black paint to eliminate light entering through the gaps on the edges. The height of the lens determines the focal distance, and we used the following formula for determining the optimal pinhole diameter for a given focal length [17].

$$D = 2\sqrt{\lambda F} \tag{1}$$

where, D is the optimal pinhole diameter, F is the focal length of pinhole setup, and λ is the wavelength of light (500 nm).

Initially, a pinhole setup of 2 mm focal length was fabricated for which the optimal diameter given by Eq. 1 is 0.01 mm. This diameter does not allow enough light to pass through and thus we increased the diameter to 0.1 mm for allowing light at the cost of image sharpness. We performed the experiments with this setup. Next, we fabricated a setup with 4 mm focal length with 0.1 mm pinhole diameter. This has an optimal diameter very close to 0.1 mm and gives better image sharpness. This setup has narrower field of view compared to the previous setup. Fig. 3 shows the images taken with the two pinhole setups.

C. Interface

Fig. 4 shows a block diagram of the closed loop connection of the overall system that controls the RoboFly. Copper wires interface the chip to a development board with ARM Cortex M0+ micro-controller. The board is programmed to retrieve the pixel values using an analog to digital converter with 16-bit resolution. The values are sent to MATLAB running on a PC (Host PC) using USB serial communication for visual analysis. For situations in which a high frame rate is desired, such as during aggressive flight maneuvers, the chip also allows for only sampling a subset of the pixels, by quickly incrementing the selection register past the other pixels. The analog values are stored as an array in MATLAB, and converted to normalized gray-scale values which can be displayed and processed further using in-built MATLAB functions. High level commands are sent to a second PC running Simulink real-time, which generates signals for the piezo actuators.

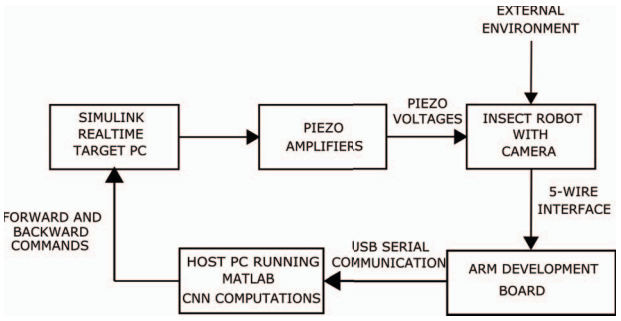

Fig. 4. The block diagram of the interface of the RoboFly with the flight weight camera mounted on it. The camera captures the images and transfers it to the host PC via a development board. The host PC performs the necessary computation and provides the control commands to a Target PC. The Target PC provides corresponding signals to the piezo actuators for the robot's motion.

III. Image Classification

To demonstrate the use of the camera to control the insect robot, we implemented an image classification task using a CNN. Using a neural network learning method for classification helps in countering image noise and serves as a test case for tackling ill-defined identification tasks for which they are suited [18] [19]. Pictures of three different flowers and predators were placed in front of the camera. The pictures were captured using both the pinhole setup. We first used the images captured with the pinhole setup of 2 mm focal length and performed the experiments as explained in the experiments section. A shallow CNN with just one convolution layer was used for classifying the images into two classes, either predators or flowers. The layers of the CNN are as follows,

1) A 2-D image input layer that receives a raw captured image as an input to the CNN.
2) A convolution layer of stride length 1.
3) A Rectifier Linear Unit (ReLU) layer as the activation function.
4) A maximum pooling layer with a 2 $\times$ 2 window and a stride length of 2.
5) A fully connected layer with outputs equal to number of classes.
6) A softmax layer for normalising the outputs from the fully connected layer.
7) A classification layer that classifies the image as a flower or predator.

Next, the pictures were captured with the setup of 4 mm focal length. The captured pictures are as shown in Fig. 5 and Fig. 6. The same CNN layers were used for the classification. With this setup, we get higher test classification accuracy than the previous one (95% vs 85%) using fewer learned filters (5 vs 50).

We also used the 4 mm focal length pinhole setup to classify our dataset into 6 classes (3 different flowers and 3 different predators). The subsequent sections give details of the training and test classification accuracy of the CNN for this task.

A. Training the CNN

The CNN was trained with black images printed on a white sheet. A light source was placed in front of the sheet and the reflection of the image was captured by the camera. The gray-scale imagery reduces the number of channels required for the CNN thereby decreasing the computation to a third of what it will take for a RGB image. Each image was

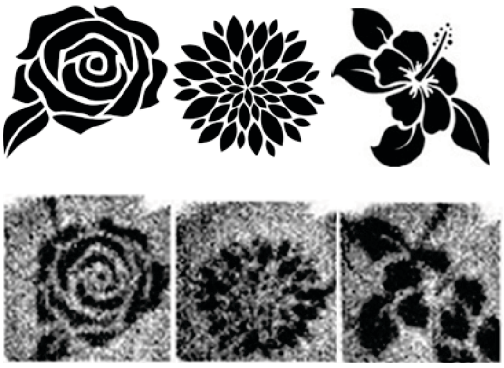

Fig. 5. Black images of the three flowers used as the samples for classification task printed on a white sheet (top). The corresponding images of the flowers as captured by the camera with pinhole setup of 4 mm focus, and used for training the CNN (bottom)

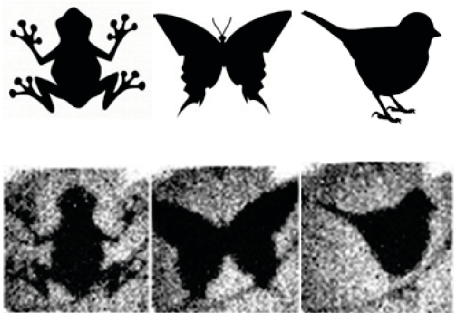

Fig. 6. Black images of the three predators used as the samples for classification task printed on a white sheet (top). The corresponding images of the predators as captured by the camera with pinhole setup of 4 mm focus, and used for training the CNN (bottom)

captured by varying illuminations and light source angles. The obtained images were also rotated to different angles to overcome rotational variance. A total of 1080 images were generated as training data. The images were taken in the raw form without any filters for noise correction and used for training the CNN. The CNN was trained using back propagation [20] with stochastic gradient descent algorithm with a learning rate of 0.001. All the operations were performed in MATLAB using the Neural Network toolbox. Another set of 360 images of the same flowers and predators were captured and used as test data for testing the accuracy of the CNN. Fig 7 shows the training and test classification accuracy for different number of learned filters for window sizes of 3, 4 and 5.

B. Computational Constraints

Implementing neural networks on insect scale robots give us insights into how insect nervous systems might use visual feedback to control flight, a process that is still not fully understood. The main constraints are the computational expense when implementing these tasks with on-board microcontrollers. For future on-board processing, we target an ARM Cortex M4 class processor, which is available at clock speeds up to 120 MHz in an 8 mg Wafer Level Chip Scale Package (WLCSP) with 512 kB flash memory and 176 kB RAM (Atmel SAMG5).

The micro-controller includes Digital Signal Processing (DSP) resources like a 32 bit, 1 cycle Multiply and Accumulate (MAC) unit. The main multiplication and addition tasks for the CNN are the convolution layer and the fully connected layer. Other operations include comparing and storing values in the ReLU and maximum pooling layers. We assume that pre-fetch libraries will be used for reading operations from flash. Thus we assume 1 cycle each for reading operations from flash and RAM. This allows us to estimate the number of cycles required for a single classification with a particular number of learned filters and convolution window size as shown in Eq. 2.

$$\begin{aligned} L1 &= p^2 \times K^2 \times f + p^2 \\ L2 &= p^2 \times f \\ L3 &= (p/2)^2 \times 3 \times f + (p/2)^2 \times f \\ L4 &= (p/2)^2 \times N \times f + (p/2)^2 \times f \\ Total\ cycles &= L1 + L2 + L3 + L4 \end{aligned} \tag{2}$$

where $L1$, $L2$, $L3$, $L4$ are the cycles required for convolution, ReLU activation, max pooling and fully connected layers respectively; f is the number of learned filters in the convolution layer; p is the number of pixels along the side of the square pixel array; K is the convolution window size; and N is the number of classes in the fully connected layer. This is proportional to the number of such classifications made per second as shown in Eq. 3.

$$C \propto \frac{(1\ sec) \times (120 \times 10^6\ MHz)}{Total\ cycles} \tag{3}$$

where C is the number of classifications per second. Fig 7 shows the relationship between the number of learned filters and the number of classifications per second for window sizes of 3, 4, and 5.

TABLE I. Table with the RAM and Flash Memory requirements for the CNN layers

Layer	Flash (kB)	RAM (kB)
Convolution Layer	2	0
Max Pooling Layer	0	100
Fully Connected Layer	480	0
Total Memory	482	100

We assume that the training will be performed offline. The main on-board storage requirement is for the weights of the convolution layer and fully connected layer. We assume that we perform a convolution, ReLU activation and max pooling operation simultaneously and store these into the RAM memory. Thus the RAM is utilized mostly by the output of max pooling layer. The other layers do not require significant storage and contribute only towards computation. Table I shows that flash and RAM memory used by the three layers for a convolution window size of 5 and 40 number of

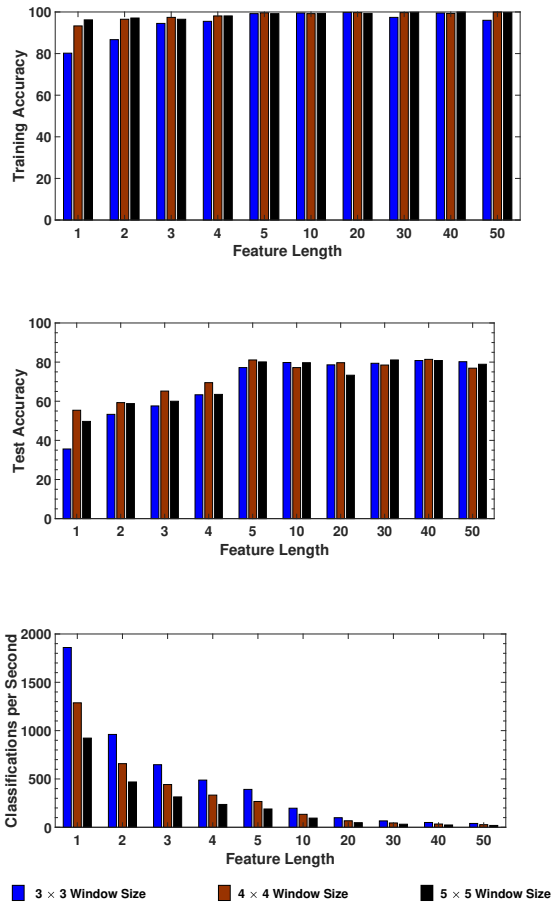

Fig. 7. Plots relating the training accuracy, test accuracy and the classifications per second for various number of learned filters (feature length) for window sizes 3,4 and 5. For all the window sizes, the training and test accuracy increased upto 5 learned filters, after which they reached an accuracy of around 99% and 80% respectively. The classifications per second decreased with increase in number of learned filters.

learned filters are compatible with the target micro-controller. More than 40 learned filters do not fit into the RAM memory. The weights are assumed to be stored at 16 bits per weight.

C. Discussions

From Fig 7 we can see that the CNN with number of learned filters of 1 to 4 do not capture all the features very well, which is evident in the training accuracy. There is a gradual increase in the test accuracy. For more than 5 learned filters, the CNN captures the features well and has training accuracy around 99% while the test accuracy reaches 77-80%. Since the amount of training data is very small, the models tend to overfit after a particular number of learned filters. Getting more comprehensive training data would provide better performance, but that is not the emphasis for this work.

The number of the classifications per second goes down as the number of learned filters increases. The latency of image capture increases the time taken for each classification. Thus there is a trade-off between the classification accuracy and the number of classifications that can be made per second. Choosing an optimal classification rate and accuracy is important for high speed tasks as mentioned in section I. The present study is concerned primarily with a system-level proof of principle that a CNN could be used on an insect-sized robot, and less concerned with the specific characteristics of our implementation. We therefore leave the analysis of the learned filters for future work.

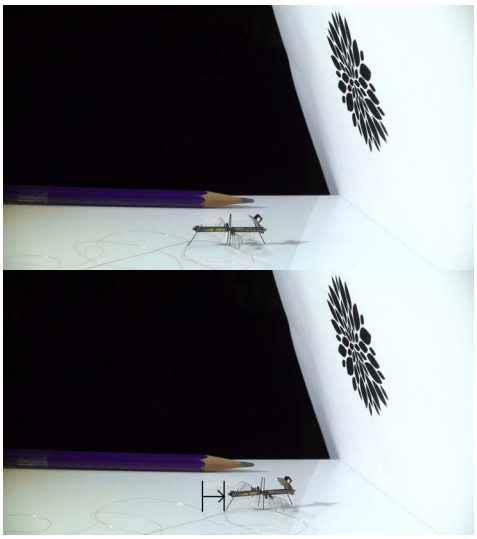

Fig. 8. When the picture of a flower is placed as shown, the trained CNN classified it as a flower and the robot moved towards it. (a) shows the initial position at time T = 0 s. (b) shows the final position at time T = 1 s. Forward motion was restricted to 1 s to avoid collision with the image sheet.

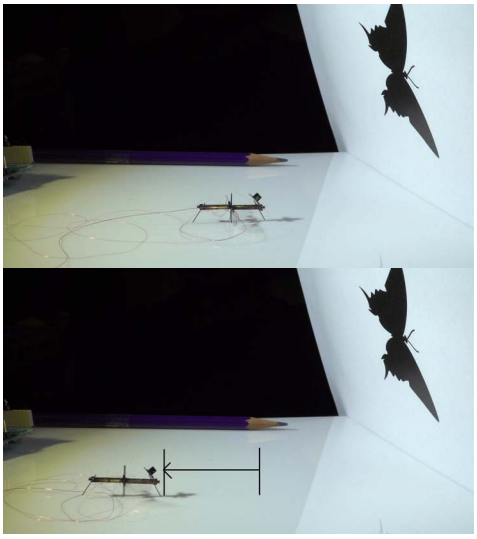

Fig. 9. When the picture of a predator is placed as shown, the trained CNN classified it as a predator and the robot moved away from it. (a) shows the initial position at time T = 0 s. (b) shows the final position at time T = 4 s.

IV. Experimental Results

We also performed a test of the camera mounted on the RoboFly. For these tests, we used an earlier design of our camera with a 2 mm focal length providing lower accuracy. The insect robot was placed in front of the images of the flowers and the predators. The onboard camera captured the images, and the trained CNN classified them.

The robot moved toward the flower images and away from the predator images based on the feedback provided by the CNN in real-time. Fig. 8 shows initial and final time instances of the insect robot moving forward toward a flower image and Fig. 9 shows initial and final time instances of the insect robot moving backward away from a predator image. Forward motion was restricted to 1 s to prevent the insect robot from colliding with the image sheet.

V. CONCLUSION AND FUTURE WORK

The paper presents the fabrication and interface of a low-weight camera onto an insect scale robot, the RoboFly. Compared to a previous implementation of a vision sensor on an insect scale robot [1], the work here increased the resolution (64×64 vs 4×32) and reduced the weight (26 mg vs 33 mg). The camera is used for implementing visual control tasks. As a demonstration, image classification using CNN is performed with the images captured by the camera to make the insect robot recognize a flower and a predator image and move toward or away.

The paper also discusses the possibility of implementing the computation on board the insect robot. Our results indicate that current ultra-light embedded micro-controllers are capable of the necessary computation at the necessary speed. The results can be seen as a step towards performing flight control tasks such as landing site detection and obstacle avoidance using only components carried on-board. We believe such tasks, which are hard to explicitly specify, are well-suited to the model-free type of operations performed by neural networks.

Future work will involve implementing flight control using optical flow. Optical flow has been used for altitude control [1], hovering [21]–[23], and landing [24]. Compared to other techniques such as feature based visual Simultaneous Localization And Mapping (SLAM), optic flow has far lower computational requirements. For example, it was shown that a hovercraft robot with fly like dynamics could visually navigate a 2-D corridor using optic flow in a way that only required 20 KFLOPS [25]. Extensions of this to 3-D should fall within the computational constraints of insect scale.

REFERENCES

[1] P.E. Duhamel, N. O. Perez-Arancibia, G. L. Barrows, and R. J. Wood, "Biologically inspired optical-flow sensing for altitude control of flapping-wing microrobots", *Mechatronics, IEEE/ASME Transactions on*, vol. 18, no. 2, pp. 556568, 2013.

[2] M. Blosch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments", in *Proc.IEEE Int. Conf. on Robotics and Automation*, 2010.

[3] R. Moore, K. Dantu, G. Barrows, and R. Nagpal, "Autonomous MAV guidance with a lightweight omnidirectional vision sensor", in *Proc.of IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014.

[4] S. Ahrens, D. Levine, G. Andrews, and J. P. How, "Vision-based guidance and control of a hovering vehicle in unknown, gps-denied environments", in *Proc. IEEE International Conference on Robotics and Automation ICRA 09*, 2009, pp. 26432648.

[5] L. Minh and C. Ha, "Modeling and control of quadrotor mav using vision-based measurement", in *Strategic Technology (IFOST), 2010 International Forum on. IEEE*, 2010, pp. 7075.

[6] F. Fraundorfer, L. Heng, D. Honegger, G. Lee, L. Meier, P. Tanskanen, and M. Pollefeys, "Vision-Based Autonomous Mapping and Exploration Using a Quadrotor MAV", in *Intelligent Robots and Systems (IROS), 2012 IEEEIRSJ International Coriference on*, oct 2012.

[7] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments", in *Proc. SPIE Unmanned Systems Technology XI*, 2009.

[8] A. T. Singh, Y. M. Chukewad, and S. B. Fuller, "A robot fly design with a low center of gravity folded from a single laminate sheet", in *workshop on Folding in Robotics, IEEE conference on Intelligent Robots and Systems* (2017).

[9] J. James, V. Iyer, Y. Chukewad, S. Gollakota, S. B. Fuller, "Liftoff of a 190 mg Laser-Powered Aerial Vehicle: The Lightest Untethered Robot to Fly, in *Robotics and Automation (ICRA), 2018 IEEE Int. Conf. IEEE*, 2018.

[10] Y. M. Chukewad, A. T. Singh, and S. B. Fuller, "A New Robot Fly Design That is Easy to Fabricate and Capable of Flight and Ground Locomotion, in *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on. IEEE*, 2018. (accepted)

[11] K. Ma, S. Felton, and R. Wood, "Design, fabrication, and modeling of the split actuator microrobotic bee", in *Intelligent Robots and Systems(IROS), IEEE/RSJ International Conference on. IEEE*, 2012.

[12] V. Kumar and N. Michael, "Opportunities and challenges with autonomous micro aerial vehicles", *Int. J. Robot. Res. (IJRR)*, vol. 31,no. 11, pp. 12791291, 2012.

[13] S. Lupashin, A. Schollig, M. Sherback, and R. D. Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips", in *Proc. of the IEEE Int. Conf. on Robotics and Automation, Anchorage, AK*, May 2010, pp. 16421648.

[14] G. P. Subramanian, "Nonlinear control strategies for quadrotors and CubeSats", *Masters thesis, University of Illinois at Urbana-Champaign*, 2015

[15] D. C. Ciresan, U. Meier, and J. Schmidhuber,"Multi-column Deep NeuralNetworks for Image Classification", in *Proceedings of Proceedings of Computer Vision and Pattern Recognition*, 2012, pp. 3642-3649.

[16] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection", in *IEEE Patt. Anal. Mach. Intell., volume 20, pages 2238*, 1998.

[17] J. W. Strutt, "On Pin-hole Photography", *Phil. Mag., v.31, pp 87-99*, 1891.

[18] M. E.-Petersen, D. de Ridder, and H. Handels, "Image processing with neural networksa review", *Pattern Recognition, vol. 35, pp.22792301*, 2002.

[19] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks", in *NIPS*, 2012.

[20] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation", in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, D. E. Rumelhart (AAAI-91), July 1991, pp. 762-767. and James L. McClelland, Eds., vol. 1, ch. 8, pp. 318-362. Cambridge, MA: MIT Press*, 1986.

[21] D. Honegger, L. Meier, P. Tanskanen, and M. Pollefeys, "An open source and open hardware embedded metric optical flow cmos camera for indoor and outdoor applications", in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2013.

[22] V. Grabe, H. H. Bulthoff, D. Scaramuzza, and P. R. Giordano, "Nonlinear ego-motion estimation from optical flow for online control of a quadrotor UAV", *The International Journal of Robotics Research, vol. 34, no. 8, pp. 11141135*, 2015.

[23] S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart, "MAV navigation through indoor corridors using optical flow", in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2010.

[24] B. Herisse, F.-X. Russotto, T. Hamel, and R. Mahony, "Hovering flight and vertical landing control of a vtol unmanned aerial vehicle using optical flow", in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems IROS , pp. 801806*, 2008.

[25] S. B. Fuller and R. M. Murray, "A hovercraft robot that uses insect inspired visual autocorrelation for motion control in a corridor", in *IEEE International Conference on Robotics and Biomimetics (ROBIO), Karon Beach, Phuket, pp. 14741481*, 2011.