SWARM INTELLIGENCE:

Micro-Mechanical Flying Insect for Low Cost Outdoor Surveillance

Subhadip Mitra

S. Ravi Kumar, Rajneesh, Rohit Khansili

Technical Proposal

Submitted to

Defence Research and Development Organisation (DRDO)

Ministry of Defence, Government of India

Golden Jubilee Nationwide Students' Competition 2008

Amity School of Engineering

Amity University, Uttar Pradesh

2007-2008

ABSTRACT

This paper proposes a swarm-based micro-mechanical flying insect (MFI) system for outdoor surveillance applications. The idea is simple but powerful: instead of one expensive UAV, we deploy hundreds of tiny insect-sized flyers that work together like a swarm of bees. Each unit costs very little to manufacture, and even if some are lost, the swarm keeps functioning.

The flying mechanism uses piezoelectric actuators for the main wing beating (similar to what researchers at UC Berkeley have demonstrated) combined with shape memory alloy muscle wires for steering and pitch control. For sensing, we have designed an artificial compound eye based on insect vision, which gives wide field of view in a very small package. The real contribution of this work lies in three algorithms we have developed: (1) the Adaptive Weighted Covering Graph that tells the swarm which areas need watching and when, (2) the Multi-State Flag Matrix that lets multiple flyers coordinate without any central controller, and (3) the TSIGO algorithm that makes smart use of the limited processing power available for the compound eye.

We believe this system can be built with current technology and will be useful for border surveillance, urban monitoring, and search operations. The cost per unit is expected to be under Rs. 5000 in mass production.

Keywords: Micro-mechanical flying insect, swarm intelligence, piezoelectric actuators, muscle wire, compound eye, surveillance, MAV, distributed coordination

1. INTRODUCTION

1.1 Background

India has over 15,000 km of land borders and 7,500 km of coastline to monitor. The current approach of using manned patrols, watchtowers, and occasional UAV flights is expensive and has gaps in coverage. A soldier cannot be everywhere at once, and even our Lakshya and Nishant UAVs cost crores of rupees each.

Nature has already solved this problem. A swarm of bees can monitor a large area, communicate threats to each other, and continue operating even when individual bees are lost. They do this with brains smaller than a grain of rice. If we can build machines that work on similar principles, we can change how surveillance is done.

The idea of building flying robots at insect scale is not new. DARPA in USA started their MAV programme in 1997, and Prof. Fearing's group at UC Berkeley has been working on the Micromechanical Flying Insect since the late 1990s. What we are proposing is to take this technology and add proper swarm intelligence algorithms so that hundreds of these flyers can work together as a team.

1.2 Why This Matters

Consider the situation at the India-Pakistan border or in the Northeast. Traditional surveillance has three problems. First, it is expensive - a Searcher UAV costs around Rs. 10 crore. Second, these systems can be detected and shot down. Third, they cannot go inside buildings or dense forest.

An insect-sized flyer solves all three problems. It is so cheap that losing a few does not matter. It is so small that it is nearly impossible to detect or shoot. And it can fly through windows, into tunnels, through forest canopy - places no other surveillance platform can reach.

1.3 Objectives

We aim to achieve the following:

1. Design a flying platform using piezo actuators and muscle wire that can actually fly
2. Build a compound eye sensor that gives useful imagery in a package under 30 mg
3. Develop algorithms for swarm coordination that work without any central computer
4. Integrate GPS (on relay nodes) and wireless imaging for real operations
5. Demonstrate the system working in a controlled outdoor test

1.4 Our Contributions

The main original work in this project is in the algorithms. Many groups are working on the hardware - we have studied their designs and adapted them. But nobody has properly solved the coordination problem for a swarm of such tiny flyers. We propose three new ideas:

- **Adaptive Weighted Covering Graph (AWCG):** A way to represent the surveillance area where priority automatically increases for places that haven't been checked recently. When a threat is detected, nearby areas also get higher priority. This is inspired by how security guards actually patrol - they check important spots more often and rush to areas where something suspicious is reported.
- **Multi-State Flag Matrix (MSFM):** A method for multiple flyers to divide up the surveillance area without fighting over the same spot. The clever part is a hash function that lets each flyer independently calculate who should go where, so they all agree without needing to discuss it.
- **Temporal-Spatial Information Gain Optimization (TSIGO):** An algorithm for processing compound eye data efficiently. Since we have very limited computing power on board, we cannot analyze all pixels equally. TSIGO figures out which parts of the image are most likely to contain useful information and focuses processing there.

2. PREVIOUS WORK

2.1 Micro Flying Robots

The pioneering work in this field comes from Prof. Ron Fearing's lab at UC Berkeley. Their Micromechanical Flying Insect project started in 1998 and has demonstrated that insect-scale flight is achievable with man-made machines. They use a four-bar mechanism in the thorax to convert small piezo displacements into large wing rotations - the same trick that real flies use.

Prof. Dickinson at Caltech has done excellent work on understanding how insect flight actually works. His 1999 Science paper showed that insects use unsteady aerodynamics - delayed stall, rotational lift, and wake capture - to generate much more lift than you would expect from their small wings. This is important for us because it means we need to replicate these complex wing motions, not just simple flapping.

More recently, Prof. Robert Wood at Harvard has demonstrated a 60 mg robotic fly that can actually take off and hover. His work on piezo bimorphs and ultra-light carbon fibre structures is directly relevant to our design.

2.2 Actuation Methods

For flapping at 150-250 Hz (typical insect wing beat frequency), piezoelectric actuators are the only practical option today. PZT ceramics can switch in microseconds and have good power density. The Berkeley group has shown actuators that produce enough force and displacement for flight.

Shape memory alloys like Nitinol are too slow for primary flapping - they take tens of milliseconds to actuate. But they are excellent for secondary control because they have very high force-to-weight ratio. We plan to use SMA muscle wires (like Flexinol from Dynalloy) for steering and wing pitch adjustment.

2.3 Swarm Intelligence

The book by Bonabeau, Dorigo and Theraulaz (1999) is the standard reference for swarm intelligence. The key insight is that complex group behaviour can emerge from simple individual rules. Each ant in a colony follows basic rules about pheromones, yet the colony as a whole can find shortest paths and allocate workers efficiently.

For robotics, Reynolds' boids model (1987) showed that flocking behaviour needs just three rules: don't crash into neighbours, fly in roughly the same direction, and stay with the group. We build on this but add specific rules for surveillance - who covers which area, how to respond to threats, etc.

2.4 What Is Missing

Looking at all this previous work, we found three gaps. First, existing coverage algorithms treat all areas equally - they do not understand that surveillance priority changes with time (an area checked 1 hour ago is more urgent than one checked 5 minutes ago). Second, most multi-robot coordination schemes need either a central computer or complex voting among robots. This does not work when you have hundreds of tiny flyers with minimal communication bandwidth. Third, nobody has addressed how to efficiently process compound eye imagery with the extremely limited computing available on a 100 mg platform.

These gaps are what we address in this project.

3. SYSTEM DESIGN

3.1 Overall Architecture

The complete system has three types of nodes: the MFI flyers themselves, ground relay stations, and a base station. The MFI flyers do the actual surveillance. Ground stations carry heavier equipment like GPS receivers and more powerful radios, and also serve as perching points for MFIs to save energy. The base station is where human operators monitor the feeds.

3.2 MFI Platform Specifications

We are targeting the following specifications based on what the Berkeley and Harvard groups have achieved:

- Total mass: 100-150 mg
- Wingspan: 20-25 mm (comparable to a housefly)
- Wing beat frequency: 150-250 Hz

- Payload for sensors and electronics: 20-30 mg
- Power for hovering: estimated 8-10 mW

The frame is built from carbon fibre composite using laser cutting and folding techniques (the Smart Composite Microstructures method from Berkeley). This gives us a strong, light structure that can be made in batches.

3.3 Actuation System

3.3.1 Piezoelectric Thorax

The wings are driven by PZT unimorph actuators in a bimorph configuration. When voltage is applied, the piezo bends, and this motion is amplified by the four-bar thorax mechanism to produce the large wing strokes needed for flight. The system is driven at its resonant frequency to minimize power consumption - at resonance, you get maximum displacement for minimum electrical input.

The driving signal comes from a small oscillator circuit. We plan to use a PIC microcontroller (PIC12F series) running at 4 MHz for this, which also handles the basic control logic. Total power for the electronics should be under 1 mW.

3.3.2 SMA Muscle Wires

For steering and attitude control, we use 37 micron diameter Flexinol wire arranged in opposing pairs. When you pass current through the wire, it heats up and contracts. By activating the left or right wire, we can tilt the wing roots and change the direction of flight.

The response time is about 20-50 ms for contraction. Cooling takes longer (relies on air convection), but during flight the airflow speeds this up considerably. For basic manoeuvres like turning and altitude changes, this response is adequate.

4. COMPOUND EYE SENSOR AND TSIGO ALGORITHM

4.1 Why Compound Eye?

A regular camera lens is too heavy for our platform. But we can make an artificial compound eye using a microlens array - many tiny lenses, each focusing onto one pixel. This gives a wide field of view (we are targeting 180 degrees) in a package weighing under 15 mg. The resolution is low (maybe 64 to 256 "pixels"), but insects do amazingly well with similar vision.

We fabricate the microlens array by hot embossing PMMA sheets - this is a standard MEMS technique. Each lens is about 100 microns in diameter. The photodetectors underneath are standard CMOS sensors, just very small ones.

4.2 The Processing Problem

Even with only 64 ommatidia (compound eye units), processing all of them in detail is not possible with our limited computing. We have maybe a few hundred microamps of current budget for computation. So we need to be clever about what we process.

This is where TSIGO comes in. The idea is borrowed from how human attention works - you cannot see everything in detail, so your brain focuses processing power on the parts that matter.

4.3 TSIGO: Temporal-Spatial Information Gain Optimization

4.3.1 Basic Concept

We treat processing power as a scarce resource that must be allocated wisely. We define an "attention budget" B that represents total processing available per frame. Each ommatidium can receive attention from 0 (just check for motion) to 1 (full detailed analysis). The total attention allocated cannot exceed B.

The question is: which ommatidia deserve attention? Our answer: the ones that will give us the most information about potential targets.

4.3.2 Two-Stage Processing

Stage 1 is cheap and runs on all ommatidia. We just check if the intensity changed from last frame:

$$\Delta y\_n = |y\_n(now) - y\_n(last\ frame)|$$

If Δy_n is above a threshold, something moved in that direction. This costs almost nothing - just a subtraction and comparison.

Stage 2 is expensive and runs only on selected ommatidia. Here we do actual pattern matching, edge detection, and feature extraction to figure out if the moving thing is a person, vehicle, or just a bird. This is where the attention budget gets spent.

4.3.3 Predictive Attention

The clever part is that we do not just react to motion - we predict where targets will be. If we were tracking a person walking left, we allocate attention to the ommatidia on the left where we expect them to appear next. This compensates for processing delays.

We use a simple motion model for prediction. If a target was at position x with velocity v, we guess it will be at $x + v \cdot \Delta t$ next frame. The predicted position maps to certain ommatidia, which get bonus attention allocation.

4.3.4 The Algorithm

Here is TSIGO step by step:

```
ALGORITHM 1: TSIGO

Input: current image y(k), previous targets, budget B

STAGE 1: Motion detection (runs on all ommatidia)
  for each ommatidium n:
    delta[n] = |y[n] - y_prev[n]|
    motion[n] = 1 if delta[n] > THRESH else 0

PREDICTION: Where will targets be next?
  for each tracked target j:
    predicted_pos[j] = pos[j] + velocity[j] * dt
    predicted_ommatidia[j] = which ommatidia see that pos
```

```
PRIORITY CALCULATION:
  for each ommatidium n:
    priority[n] = alpha * motion[n]
    for each target j:
      if n in predicted_ommatidia[j]:
        priority[n] += beta * importance[j]

STAGE 2: Allocate attention budget
  sort ommatidia by priority (highest first)
  remaining = B
  for each ommatidium n in sorted order:
    attention[n] = min(1, remaining)
    remaining = remaining - attention[n]
    if remaining <= 0: break

PROCESS: Run detailed analysis on high-attention ones
UPDATE: Refine target position estimates
```

Stage 1 is O(N) and fully parallel - can be done in hardware. Stage 2 is $O(B \times C)$ where C is the cost of detailed analysis. Since $B << N$ (maybe $B = 8$ when $N = 64$), this is fast enough for real-time use.

5. ADAPTIVE WEIGHTED COVERING GRAPH

5.1 The Problem

Imagine you are a security guard patrolling a building. You need to check all areas, but some areas matter more than others (the vault vs. the broom closet). Also, if you checked the vault 5 minutes ago, it is less urgent than if you checked it 2 hours ago. And if the silent alarm goes off in one wing, you want to rush there.

This is exactly the problem our swarm faces. We need to cover the surveillance area, but intelligently. Standard coverage algorithms do not account for time - once an area is "covered", they ignore it. We need something better.

5.2 The Graph Model

We divide the surveillance area into cells and represent it as a graph $G = (V, E)$. Each cell is a vertex. Edges connect adjacent cells (where an MFI can fly from one to another).

The key innovation is that each vertex has a weight that changes over time. The weight represents "how urgently does this cell need checking?"

5.3 Staleness Decay

Each vertex remembers when it was last observed (timestamp τ). The weight increases as time passes since last observation:

$$W(v, t) = W_0(v) \times [1 + \lambda \times (1 - e^{(-\kappa(t - \tau(v)))})]$$

Here W_0 is the base importance (higher for critical areas), λ controls how much the weight can increase (we use $\lambda = 2$, so weight can triple), and κ controls how fast (we tune this based on mission - faster for high-threat areas).

This formula has a nice property: weight grows quickly at first when an area goes stale, then levels off. This matches intuition - checking an area after 10 minutes vs. 5 minutes matters a lot, but after 2 hours vs. 2 hours 5 minutes, not so much.

5.4 Threat Propagation

When an MFI detects something interesting at vertex v*, we want other MFIs to come help investigate. We do this by increasing the weights of nearby vertices:

$$\Delta W(v) = A \times e^{(-distance(v, v^*)/\sigma)} \times threat\_level(v^*)$$

The weight boost is highest at the threat location and falls off with distance. This creates a kind of gravitational pull - MFIs are attracted to where the action is.

5.5 Patrol Algorithm

A single MFI chooses where to go next based on value-for-cost ratio. If I am at position p, I look at all reachable vertices and pick:

$$next\ vertex = argmax\ [W(v, t) / cost(p \rightarrow v)]$$

where cost includes energy to fly there and any risk along the way. This is a greedy algorithm - it does not plan ahead - but with staleness decay, it naturally creates good patrol patterns without explicit route planning.

ALGORITHM 2: AWCG Patrol (single MFI)

```
1. Initialize timestamps: tau[v] = -infinity for all v
2. while on_mission:
3.   observe current position, update tau[current] = now
4.   if threat_detected:
5.     broadcast alert (others will update weights)
6.   for each reachable vertex v:
7.     compute W(v, now) using staleness formula
8.     compute cost to reach v
9.   next = vertex with highest W/cost ratio
10.  fly to next
```

6. MULTI-STATE FLAG MATRIX

6.1 The Coordination Problem

If we just run AWCG independently on each MFI, they will all go to the same high-priority areas and leave other areas uncovered. We need coordination. But we cannot have a central computer telling everyone what to do - too much communication, and if the central computer fails, everything fails.

Our solution is the Multi-State Flag Matrix. It is a distributed algorithm where each MFI maintains its own copy of an assignment table and they all converge to the same assignments using only local communication.

6.2 The Flag Matrix

The matrix F has n rows (one per MFI) and m columns (one per vertex/cell). F[i][j] indicates the relationship between MFI i and vertex j:

Value	State	Meaning
0	UNCLAIMED	Nobody is responsible for this vertex
1	CLAIMED	I am responsible but not there yet
2	ACTIVE	I am there right now, watching
3	HANDOFF	Transferring responsibility to someone else

6.3 Resolving Conflicts: The Priority Function

What if two MFIs both want the same vertex? We need a way to decide who wins. Our priority function is:

$$P(i, j) = w1/distance(i,j) + w2\times energy(i) + w3\times capability(i,j) + w4\times hash(i,j)$$

The first three terms make sense: closer MFIs win, healthier MFIs win, MFIs better suited for that area win. The fourth term is the trick that makes everything work.

6.4 The Hash Trick

The hash function looks like this:

$$hash(i, j) = ((i \times 104729) \text{ XOR } (j \times 224737)) \bmod 100000 / 100000$$

The numbers 104729 and 224737 are prime. This produces a pseudo-random number between 0 and 1 that depends only on MFI id and vertex id.

The magic is this: every MFI can compute this hash independently. If MFI #5 and MFI #12 both want vertex #37, they will both compute the same priorities and reach the same conclusion about who wins. No negotiation needed, no voting, no central arbiter. They just both do the math and get the same answer.

6.5 The Protocol

The algorithm runs in synchronized rounds. Each round has three phases:

ALGORITHM 3: MSFM Protocol (each MFI runs this)

```
CLAIM PHASE:
  look for unclaimed vertices in my range
  if found: pick highest W(v), broadcast my claim + priority

RESOLUTION PHASE:
  collect claims from neighbours (within radio range)
  for each vertex with multiple claims:
    winner = whoever has highest priority
    if I am winner: set F[me][v] = CLAIMED

EXECUTION PHASE:
  fly towards my highest-priority claimed vertex
  if I arrive: set F[me][v] = ACTIVE
  if I leave: set F[me][v] = CLAIMED
  if low energy or better MFI available: set F[me][v] = HANDOFF
```

6.6 Why It Works

We can prove that this protocol converges. Each round, at least one contested vertex gets resolved (because priorities give a strict ordering). There are m vertices total. So after at most m rounds, all vertices are assigned. In practice it converges much faster because most vertices are not contested.

7. PUTTING IT ALL TOGETHER

The three algorithms form a hierarchy:

1. **AWCG** manages the surveillance area model. It decides which areas are important based on staleness and threat reports.
2. **MSFM** divides these areas among MFIs so everyone has a job and nobody fights over the same spot.
3. **TSIGO** runs on each MFI to make the most of its limited sensing capability.

Information flows both ways. When TSIGO detects a threat, this triggers AWCG weight updates, which changes MSFM assignments. So the swarm automatically concentrates on areas of interest.

8. SUPPORTING SYSTEMS

8.1 GPS and Positioning

A GPS receiver is too heavy for an individual MFI (even the smallest ones like u-blox NEO are 2-3 grams). So we put GPS on the ground relay stations, and MFIs get position information from these.

For relative positioning between MFIs, we use radio signal strength. This is crude but adequate for swarm coordination - we only need to know approximately where neighbours are.

8.2 Imaging and Communication

Besides the compound eye (used for navigation and target detection), we want to send actual images back to base. We use tiny CMOS cameras like those in medical endoscopes - 320×240 resolution in a package under 1mm square.

Image transmission uses a store-and-forward approach. An MFI captures an image, compresses it, and passes it to the nearest relay. The relay forwards it to the base. This works even if there is no direct path from the MFI to the base.

9. PROTOTYPE PLAN

9.1 Development Phases

1. **Phase 1 (6 months):** Get the basic hardware working. Build piezo actuators, test wing mechanisms, demonstrate flapping on a test stand. In parallel, simulate the algorithms in MATLAB.
2. **Phase 2 (6 months):** Integration. Assemble complete MFI units with sensors and electronics. First tethered flights. Port algorithms from MATLAB to embedded C for the PIC microcontroller.
3. **Phase 3 (6 months):** Free flight. Test with multiple MFIs operating as a swarm. Outdoor demonstrations in controlled environment.

9.2 Manufacturing Approach

We will use the Smart Composite Microstructures (SCM) technique from Berkeley. The structure is cut from carbon fibre laminate sheets using a laser cutter, then folded into 3D shape. This is like origami for robots. The advantage is that it can be done in batches - cut 100 sheets at once and fold them on an assembly line.

For the piezo actuators, we plan to work with BHEL or a similar organization that has PZT fabrication capability. The compound eye lenses can be made using standard MEMS hot embossing at any microfabrication facility.

10. EXPECTED OUTCOMES

10.1 What We Will Deliver

- Working MFI prototype that can fly (at least tethered, hopefully free)
- Compound eye module that detects motion and tracks targets
- MATLAB simulation demonstrating all three algorithms working together
- Embedded code for the algorithms running on PIC microcontroller
- Complete design documentation for future development

10.2 Applications

The immediate applications are military: border surveillance, urban reconnaissance, checking buildings before troops enter. But there are civilian uses too: search and rescue (finding people in collapsed buildings), wildlife monitoring, agricultural pest detection.

If we can bring the cost down to Rs. 5000 per unit in mass production (which we believe is achievable), this becomes practical for many applications where current UAVs are too expensive.

11. CONCLUSION

We have presented a design for swarm-based micro-mechanical flying insects for surveillance. The hardware is based on proven designs from Berkeley and Harvard. Our contribution is in the algorithms: AWCG for intelligent coverage, MSFM for distributed coordination, and TSIGO for efficient sensing.

We believe this technology is ready to move from the laboratory to practical applications. The algorithms we have developed solve real problems that have been blocking progress in swarm surveillance. And unlike some "blue sky" research, everything we propose can be built with existing technology.

For India, this represents an opportunity to leapfrog in surveillance technology. Instead of spending crores on conventional UAVs, we can deploy swarms that cost less and do more. We are excited about this project and confident we can make it work.

REFERENCES

[1] R.S. Fearing, K.H. Chiang, M.H. Dickinson, D.L. Pick, M. Sitti, J. Yan, "Wing transmission for a micromechanical flying insect," *IEEE International Conference on Robotics and Automation*, San Francisco, 2000.

[2] M.H. Dickinson, F.O. Lehmann, S.P. Sane, "Wing rotation and the aerodynamic basis of insect flight," *Science*, vol. 284, pp. 1954-1960, 1999.

[3] R.J. Wood, "Design, fabrication, and analysis of a 3DOF, 3cm flapping-wing MAV," *IEEE/RSJ IROS*, San Diego, 2007.

[4] M. Sitti, D. Campolo, J. Yan, R.S. Fearing, "Development of PZT and PZN-PT based unimorph actuators for micromechanical flapping mechanisms," *IEEE ICRA*, Seoul, 2001.

[5] E. Bonabeau, M. Dorigo, G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, 1999.

[6] C.W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, pp. 25-34, 1987.

[7] L.E. Parker, "Distributed algorithms for multi-robot observation of multiple moving targets," *Autonomous Robots*, vol. 12, no. 3, pp. 231-255, 2002.

[8] M.F. Land, "Visual acuity in insects," *Annual Review of Entomology*, vol. 42, pp. 147-177, 1997.

[9] Dynalloy Inc., "Flexinol technical data sheet," available at www.dynalloy.com, 2006.

[10] Microchip Technology, "PIC12F6xx data sheet," 2007.