# Design Project 2: Predator-Prey

Fast, but heavy predator
Slow, light (maneuverable) prey

What is optimal strategy for each competitor?

http://www.youtube.com/watch?v=MgVPLNu_S-w

# Design Parameters

- Predator/Prey are both particles.
- Motion is in the vertical plane
- Predator/Prey propulsion system exerts a force with arbitrary direction, and magnitude within predetermined range.
- Predator/prey experience gravity, aerodynamic drag force, and a fluctuating random force.

Predator:

Mass: $m_r = 100kg$

Max Force: $F_{r\max} = 1.3 m_r g$

Prey:

Mass: $m_r = 10kg$

Max Force: $F_{y\max} = 1.4 m_y g$

# Control code (submit this script)

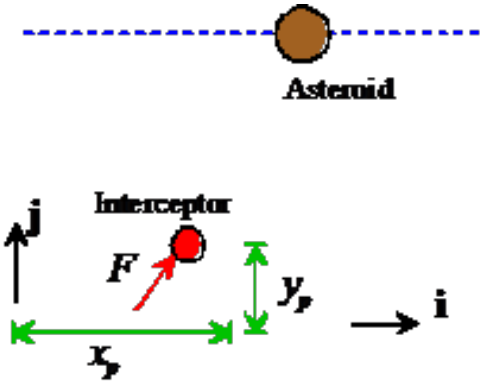
```
function F = compute_f_groupname(t,Frmax,Fymax,amiapredator,pr,vr,py,vy)
% Test time and place  Enter the time and room for your test here
% Group members:  list the names of your group members here
%   t:    Time
%   Frmax: Max force that can act on the predator
%   Fymax: Max force that can act on the prey
%   amiapredator: Logical variable - if amiapredator is true,
%              the function must compute forces acting on a predator.
%              If false, code must compute forces acting on a prey.
%   pr - 2D vector with current position of predator eg pr = [x_r;y_r]
%   vr - 2D vector with current velocity of predator eg vr= [vx_r;vy_r]
%   py - 2D vector with current position of prey py = [x_prey;y_prey]
%   vy - 2D vector with current velocity of prey py = [vx_prey;vy_prey]
%        NB:pr,vr,py,vy are all COLUMN VECTORS
%   F - 2D vector specifying the force to be applied to the object
%        that you wish to control F = [Fx;Fy]
%        The direction of the force is arbitrary, but if the
%        magnitude you specify exceeds the maximum allowable
%        value its magnitude will be reduced to this value
%        (without changing direction)

        if (amiapredator)
            % Code to compute the force to be applied to the predator
        else
            % Code to compute the force to be applied to the prey
        end
end
```

# Possible strategies

Predator:

- Apply force towards prey (see HW 3, 2012)
- Use extra credit part of HW3, 2012

$$\mathbf{n} = \left(\mathbf{r}_a - \mathbf{r} + c(\mathbf{v}_a - \mathbf{v})\right) / \left|\mathbf{r}_a - \mathbf{r} + c(\mathbf{v}_a - \mathbf{v})\right|$$




Prey:

- Apply force directly away from predator
- Move along wavy path with frequent direction changes
- Move along circular path with small radius
- Base force on *velocity* of predator in some way

# Competition rules

Each design team must play both predator and prey

Teams will compete against one another (or against random competitor)

Each contest lasts for 250 sec – if predator catches prey, predator wins.

If prey escapes, prey wins. If neither team can catch opponent, winner is determined by shortest distance between predator/prey.

**4. Scoring:**

- Each team will play the role of 'predator' and 'prey' three times, with 'blind' code (i.e. without knowing what strategy the opposing team has chosen).
- Teams will then be given 5 minutes to make any modification to their pursuit algorithm they choose, after having heard the opposing team describe their codes (the escape algorithms cannot be changed). The six tests will be repeated with the modified code.
- The overall score for each group will be determined as follows

$$S = \sum_{i=1}^{6} T_s(i) - \sum_{i=1}^{6} T_C(i)$$

where $T_S(i)$ is the time that the prey survives during the *ith* test, and $T_C(i)$ is the time taken for the predator to catch the competing prey during the *ith* test. Negative scores will be rounded up to zero. If neither team is able to catch the prey from the opposition, the winning team will be the one whose predator is able to come closest to the prey from the opposing team.

- Solutions which attempt to win or force a draw by devious means (putting MATLAB into an infinite loop, hacking the MATLAB ODE solver, etc) will be disqualified at the discretion of the judges.

# Organization

Schedule:

Project Test Date – Friday March 16th

Project is done in groups of up to 4. Can be done individually.

Deliverables:

(i) MATLAB code containing function to compute forces. Run the code through the 'function tester' script before submitting it. Due 5pm Thurs March 15th (we have to sort and test them the night before they will be run)

(ii) Report (one per group). Due Monday March 19
**Reports should be mostly written by March 16th – the extra time is so you can add a discussion of performance in the competition**

(iii) Oral presentation describing strategy only (to inform the other team how your design works) (PRESENTATIONS SHOULD NOT EXCEED 6 MINS to ensure time for the contest!)

# Reports

Reports should be written as a formal technical document.

For style guidelines read any of the papers included as background reading in earlier HW.

- Organize report carefully
- Make sure sections, paragraphs, sentences follow logical reader-friendly sequence
- Avoid colloquial language
- Avoid convoluted sentences
- Define symbols in equations, use figures.
- Figures, graphs and tables should be labeled properly and be publication quality

# Possible report outline

**Appendix 5: Report Instructions**

Your report for this project should contain the following sections:

1. *Abstract*: a few lines explaining the purpose of the report and summarizing the main conclusions
2. *Introduction*: Background information, motivation, and a brief description of the contents of the report.
3. *Pursuit strategy*: Summarize the formulas or procedures you decided to use to catch your opponent. Enough detail should be provided so that someone could duplicate your code.
4. *Escape strategy*: Summarize the formulas or procedures you decided to use to catch your opponent. Enough detail should be provided so that someone could duplicate your code.
5. *Design, and testing procedures*: Summarize candidate designs that were tested, how you made the decisions leading to your final choice, and how you tested the performance of your design.
6. **Appendix:** list any supplemental information that is useful to the reader, but not critically important to understanding how your design works. The detailed MATLAB code could go in the Appendix, for example.

# How to write a paper – Mike Ashby

To write well, you need a design. Like any design activity, there are a number of steps (Figure 1). I've used the language of engineering design here—it fits well.

**The Market Need.** What is the purpose of the document? Who will read it? How will the reader use it? The answers help you decide the length, the level of detail, the style.

**The Concept.** Good writing starts with a plan. Writers have different ways of developing plans. I find the concept-sheet (Section 3, below) is a good way to do it.

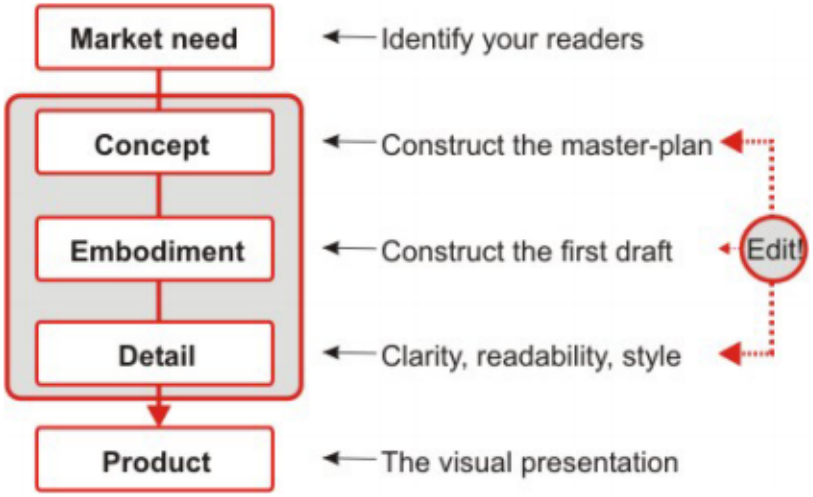



Figure 1. The Design Process. Designing a paper is like designing anything else: there are five essential steps.

**The Embodiment.** The embodiment is the first draft. Get the facts down on paper without worrying about style; make drafts of each section; develop the calculations; sketch the figures; assemble references.

**Detail.** Now comes the crafting: clarity, balance, readability; in a word —*style*.

**The End-Product.** Appearance *is* important: good layout, clear headings, well-designed figures.

The Sections that follow expand on each of these in turn.

1. Know what you want to say
2. Tell a story – use a logical order
3. Make everything as simple as possible
4. Think of your readers – can anything you say be misinterpreted? Is all the information there?
5. Label graphs, don't forget units, define symbols in equations

# Presentations

1. Keep to 7 mins (no more than 5 ppt slides) (otherwise we run out of time to test your codes!)
2. For this project, there is no need to give background (eg purpose of project, constraints etc). Focus on what your group has done.
3. Points to address: (a) What is your strategy? (b) What t are its advantages?