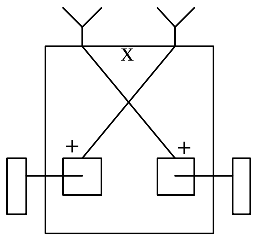
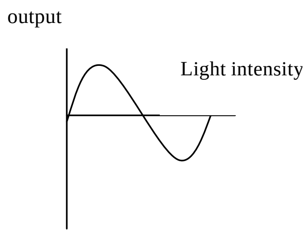
<u>31YB Tutorial 1: Thursday, 16th October, 10:00 in 2A87B</u>

Q1. Consider the following argument:
Digital computers, given sufficient memory, can simulate any process to any desired degree of accuracy.
Therefore, they can simulate the brain.
Therefore, we can produce a truly intelligent computer.
What, if anything, is wrong with the above argument.

Q2. Neural Networks are a different paradigm for computing. Discuss how does an artificial neuron model a real neuron? Try extending the argument to a network of neurons.

Q3. Consider a vehicle (in the sense of Valentino Braitenberg's book: Vehicles: experiments in synthetic perception), with the following structure:





And whose sensors have the output shown above right.
How might such vehicles behave (i) in an environment with a single vehicle and a single bright light (ii) in an environment with many such vehicles, each with a light source on the body of the vehicle, at the point marked X in the figure.

Q4. State and explain the Perceptron Learning Rule (PLR), and show using a few examples, how perceptrons can solve a linearly separable classification problem by implementing a decision surface, which is a hyper-plane of dimension one less than the dimension of the input space? What are the limitations of the PLR?

Q5. You will find some Excel spreadsheets for the perceptron learning rule, from the URL:
URL ftp://ftp.cs.stir.ac.uk) or, ftp ftp.cs.stir.ac.uk
(then login as anonymous) cd pub/staff/lss, binary, get perceptron.XLS
Download the spreadsheet, and experiment with:

1. Different initial weights
2. Different logical predicates (OR, NAND, NOR)

Does the final weight set depend on the initial weight, and does this make any difference to the perceptron's learning (performance)?

Q6. Many techniques in neural networks (and elsewhere) rely on trying to minimise some quantity. In our case, we are interested in minimising the error.
If the error is a smooth function of some variable parameters (here, the weights), then discuss how one can reduce the error by considering a 1-dimensional representation (of the weight-space)?
[NOT Examinable] Hence, show how one can use gradient descent iteratively to find the minimum of any continuously differentiable function.
[NOT Examinable] Discuss the application of the above to the general case of a high (multi)-dimensional weight space.