

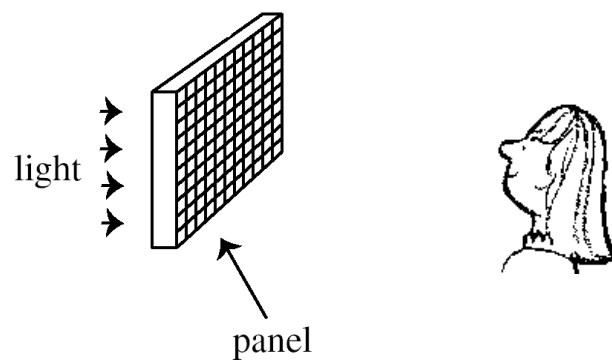
## G-I Exercises

## 1. Scanning Process with Colour LUTs

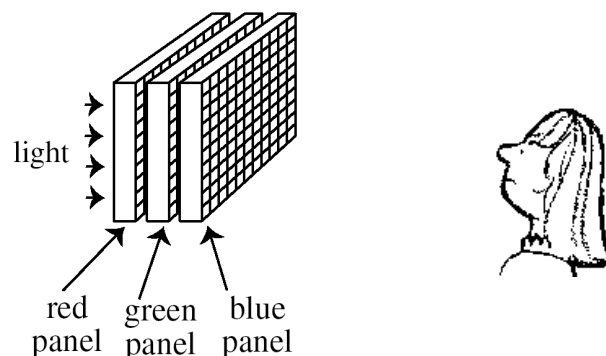
- Draw a detailed block diagram for the frame buffer scanning process where the frame buffer stores colour indices reference to a colour look-up table (LUT) for a CRT display.
- What are the limitations of this approach? Why would one want to use a colour LUT anyway? Reinforce your answer with an example for a typical graphics display with and without LUT.

## 2. An Ink Display Device

An inventor has developed a new monochrome display device: a transparent panel with a rectangular array of square pixels (see figure). The panel is tinted with a special ink which allows each pixel to range from totally transparent to transmitting only the colour of the ink. Each pixel has an 8-bit value. For example, with red ink, a pixel value of 0 would be totally transparent, 255 totally red (only red light is transmitted), and 100 a pale red.



- For a panel with  $1600 \times 1200$  pixels, how long does a pixel have to hold its state before it can be refreshed if the frame buffer memory requires 8ns to read a single byte? What is the maximal refresh rate for this setup?
- With these settings can the display operate at a resolution of  $1280 \times 1024$  and a refresh rate of 125Hz?
- The inventor found that he can make the special ink *any* colour he likes, but that each panel can be tinted with only one colour. He proposes to use three inks in three panels to make a 24-bit display: a red-tinted panel, a green-tinted panel and a blue-tinted panel will be stacked up to make a full-colour display (see figure). A value of  $(0, 0, 0)$  will be white (transparent),  $(255, 0, 0)$  red and  $(255, 255, 255)$  black.



Explain why this display does not work correctly. How can the three-panel design be modified such that it will work?

### 3. Graphics Pipeline Speedup

A simple way to reduce the number of calculations in a graphics pipeline when displaying polygonal meshes is to use a mesh primitive, such as a triangle strip. A triangle strip is a sequence of three or more vertices, in which every consecutive set of three vertices  $[(v_1, v_2, v_3), (v_2, v_3, v_4), (v_3, v_4, v_5), \text{etc.}]$  defines a triangle. Assume you intend to process  $n$  three-dimensional triangles by a graphics pipeline. What is the maximum speedup you could obtain in the front-end subsystem (transformations on the geometry) by converting triangles into triangle strips? Compute the number of multiplications and additions required to perform a transformation on a set of separate triangles and a triangle strip to support your answer.