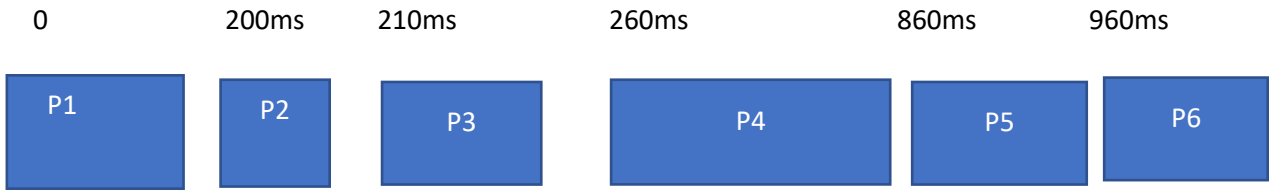
1. P1: Loads the image into the memory. (The average time is 100ms.)
   Due to the file system, the operation slows down by 500ms for each new thread (that is, one thread loads two images at 100+100ms, but two thread in parallel 100+500ms).
2. P2: Collects metadata (such as GPS coordinates) from the image. (10ms)
3. P3: Converts the image to grayscale. (50ms)
4. P4: Finds all the birds in the greyscale image. (600ms)
5. P5: Creates a new cropped image of each bird in the image. (100ms / bird)
6. P6: Saves the data (original image, metadata and new cropped images) in a database, all in one database call. (100ms)

First, we analyze existing program structure

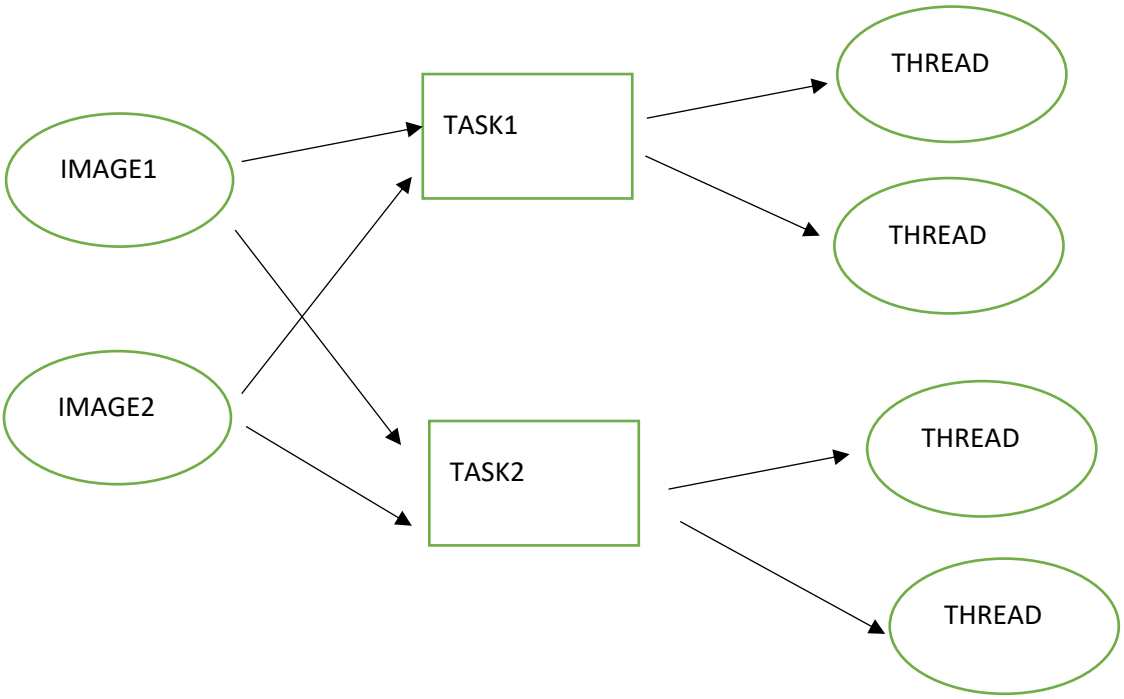
If the task is running with single-task

	CPU burst	Arrival time
P1	100+100ms	0
P2	10ms	200ms
P3	50ms	210ms
P4	600ms	260ms
P5	100ms*N	860ms
P6	100ms	860+100ms*N



But if the task running with multi-task with one core which consider as context-switch.

Next we would design a solution to run these task concurrently.



Basically, the idea is that we need to design a program that can approach parallelism with the Combination of multi-task and multi-threads.

Since the more thread we use, the more time system takes as for the OS.

Due to the file system, the operation slows down by 500ms for each new thread, so if we use 2 threads for the single task it will take 1000ms for two thread run parallel.

We divide works for 3 threads. 3 threads (each threads) start after 100+1500ms

Here is our solution for 3 threads:

The row 3 4 5 represent the thread 1 2 3 respectively. Each thread switch process between and do it synchronously, which mean process finish the next process continue.

But the efficient is not really high.

Other solution is we use 6 threads and process 5 images parallelly.

We denote that Px(y) which is x is the stage of process (out of 6) and y is the no of picture need to be processed (out of 5).For instance, P3(2) means the process stage 3 of picture 2 .

*Note: Each box accounts for 50ms

.

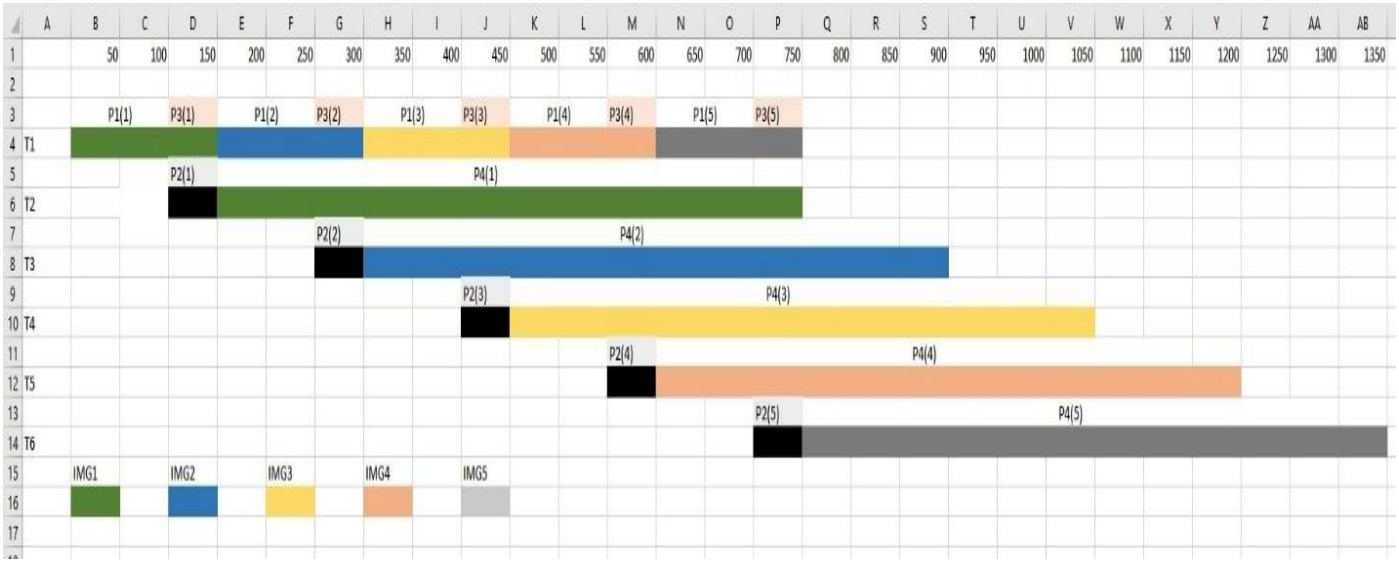


Figure 1: All thread's process from begin

Thread 1 at first is mainly used for loading image of all 5 pictures as other do their tasks independently which would first save time if all 6 thread together load image (which may takes 500ms*6) and secondly which will avoid dead-clock.

The black box is P2(Y) of each thread which takes only 10ms and it would be run parallel to P3(Y) after the P1(Y) finish. The reason we did this since P2 and P3 are independent to each other so it could run asynchronously.

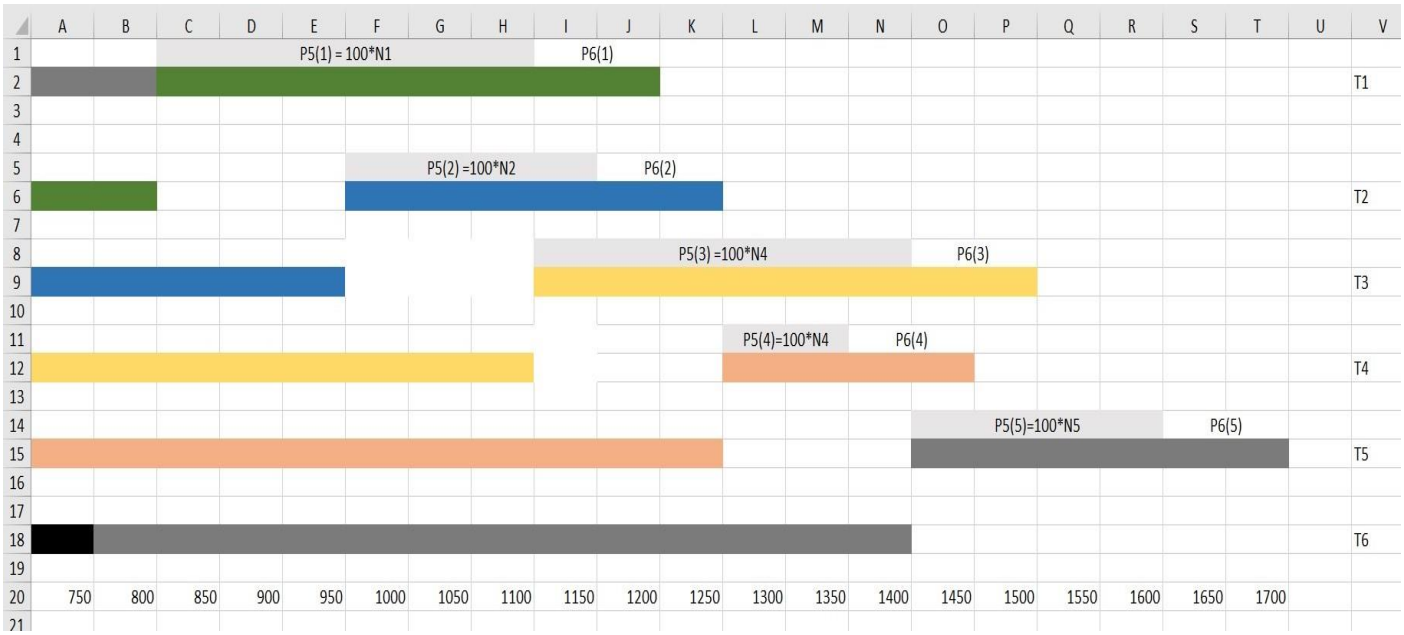


Figure 2 : All thread's processes after P3(5)

Here N1, N2, N3,.. is the number of birds in the picture 1, 2, 3 ... respectively.

In this view we denote that N1= 3, N2 = 2, N3 = 3, N4 = 1, N5 = 2.

And as the figure it will takes total of 1700ms to process each 5 pictures.