CS35L – *Winter '19*

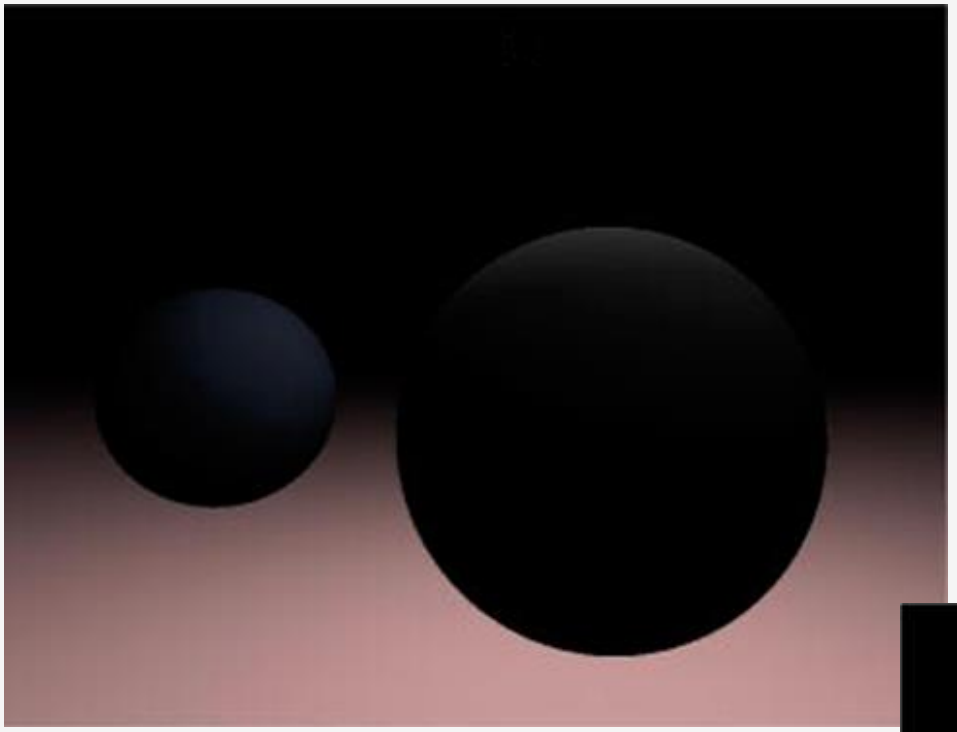
Slide set:	6.2
Slide topics:	Multithreaded programming
Assignment:	6

Ray Tracing

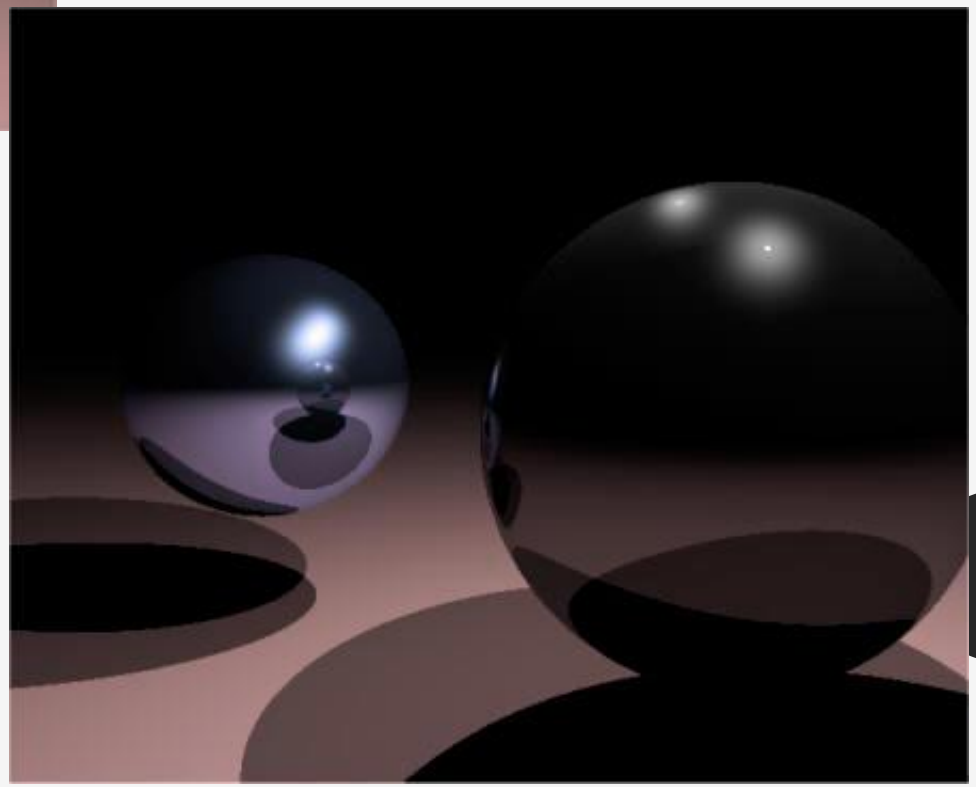
An advanced computer graphics technique for rendering 3D images

Mimics the propagation of light through objects

Simulates the effects of a single light ray as it's reflected or absorbed by objects in the images



Without ray tracing



With ray tracing

Computational Resources

Ray Tracing produces a very high degree of visual realism at a high cost

The algorithm is *computationally intensive*

Good candidate for multithreading (embarrassingly parallel)

Homework 6

Download the single-threaded ray tracer implementation

Run it to get output image

Multithread ray tracing

Run the multithreaded version and compare resulting image with single-threaded one

Basic pthread Functions

include <`pthread.h`> and link with the -`lpthread` library

There are 5 basic pthread functions

1. `pthread_create`

- creates a new thread within a process

2. `pthread_join`

- waits for another thread to terminate

3. `pthread_equal`

- compares thread ids to see if they refer to the same thread

4. `pthread_self`

- returns the id of the calling thread

5. `pthread_exit`

- terminates the currently running thread

pthread_create

Function: creates a new thread and makes it executable

Can be called any number of times from anywhere within code

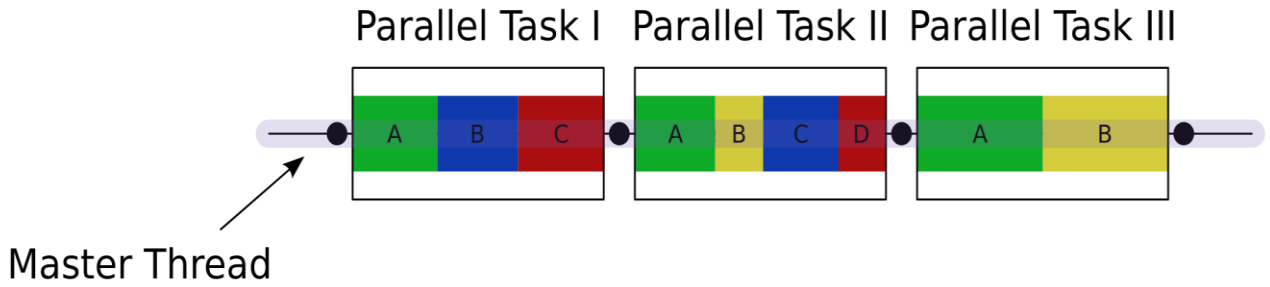
Return value:

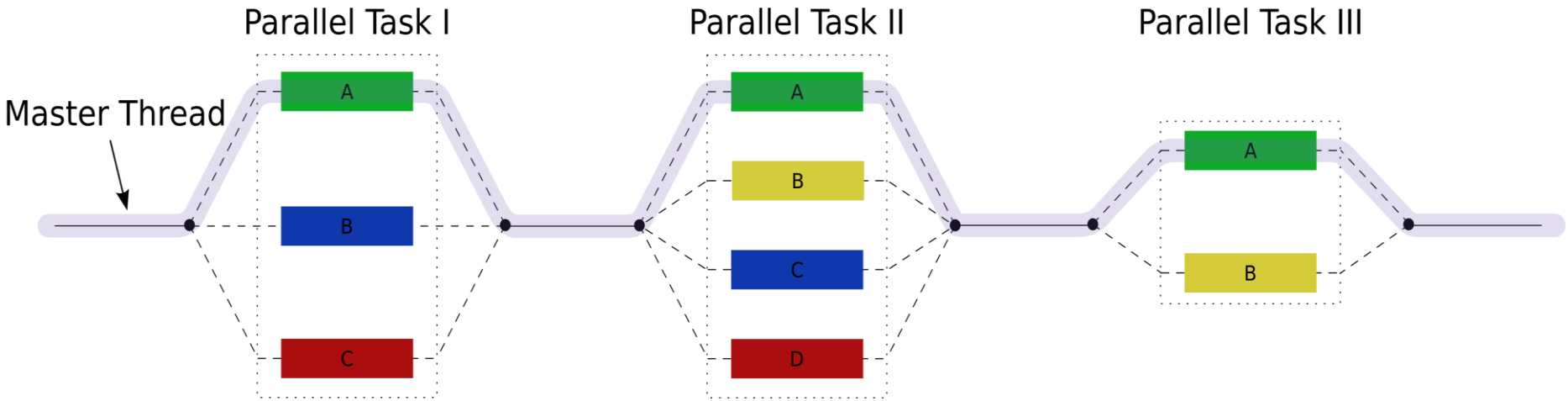
`Success: 0`	`Failure: error number`

```
int pthread_create( pthread_t *tid, const pthread_attr_t *attr,
void *(run_function)(void *), void *arg );
```

Parameters

- **tid**: unique identifier for newly created thread
- **attr**: object that holds thread attributes (priority, stack size, etc.)
- Pass in NULL for default attributes
- **run_function**: function that thread will execute once it is created
- **arg**: a *single* argument that may be passed to my_function
- Pass in NULL if no arguments





THE FORK-JOIN MODEL

pthread_create Example

```
#include <pthread.h> …
void *printMsg(void *thread_num) {
       int t_num = (int) thread_num;
       printf("It's me, thread #%d!\n", t_num);
}

int main() {
       pthread_t tids[3];
       int t;
       for(t = 0; t < 3; t++) {
              ret = pthread_create(&tids[t], NULL, printMsg, (void *) t);
              if(ret) {
                     printf("Error creating thread. Error code is %d\n", ret");
                     exit(-1);
              }
       }
}
```

Possible problem with this code?

If main thread finishes before all threads finish their job -> incorrect results

pthread_join

Makes originating thread wait for the completion of all its spawned threads' tasks

Join	
Without join, the originating thread would exit as soon as it completes its job	A spawned thread can get aborted even if it is in the middle of its chore

Return value	
`Success: 0`	`Failure: error number`

Arguments

```
int pthread_join(pthread_t tid, void **status);
```

- **tid**: thread ID of thread to wait on
- **status:** the exit status of the target thread is stored in the location pointed to by *status
  - Pass in NULL if no status is needed

pthread_join Example

```
#include <pthread.h> …
#define NUM_THREADS 5

void *PrintHello(void *thread_ num) {
        printf("\n%d: Hello World!\n", (int) thread_num);
}

int main() {
        pthread_t threads[NUM_THREADS];
        int ret, t;
        for(t = 0; t < NUM_THREADS; t++) {
                printf("Creating thread %d\n", t);
                ret = pthread_create(&threads[t], NULL, PrintHello, (void *) t);
                // check return value for errors
         }

         for(t = 0; t < NUM_THREADS; t++) {
                ret =  pthread_join(threads[t], NULL);
                // check return value for errors
         }
}
```

Homework 6

Build

Build a multi-threaded version of Ray tracer

Modify

Modify "main.c" & "Makefile"

- Include <pthread.h> in "main.c"
- Use "pthread_create" & "pthread_join" in "main.c"
- Link with –lpthread flag (LDLIBS target in Makefile)

Make

make clean check

- Outputs "1-test.ppm"
- Can see "1-test.ppm" in GIMP/Image viewer

baseline.ppm & 1-test.ppm