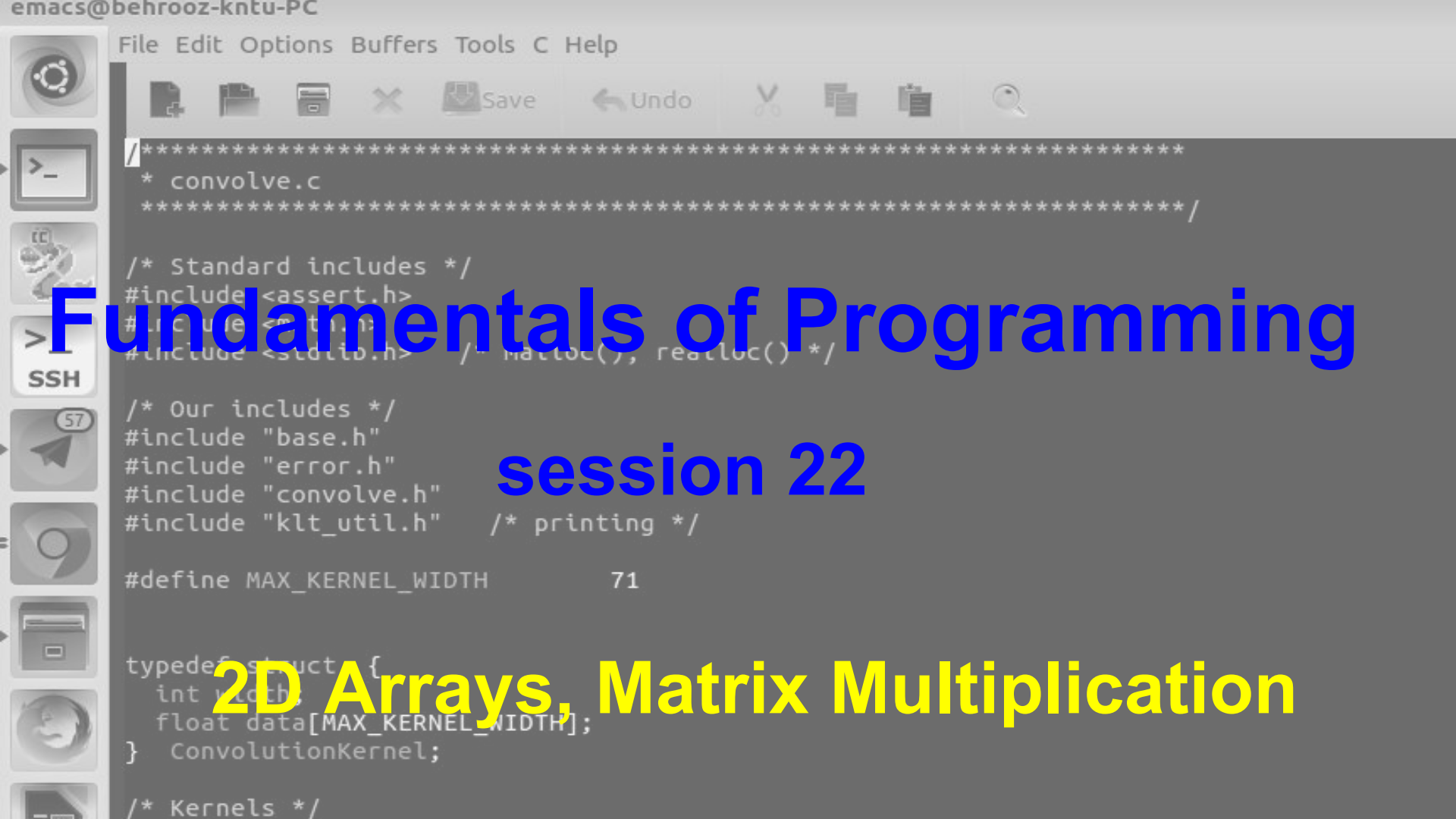
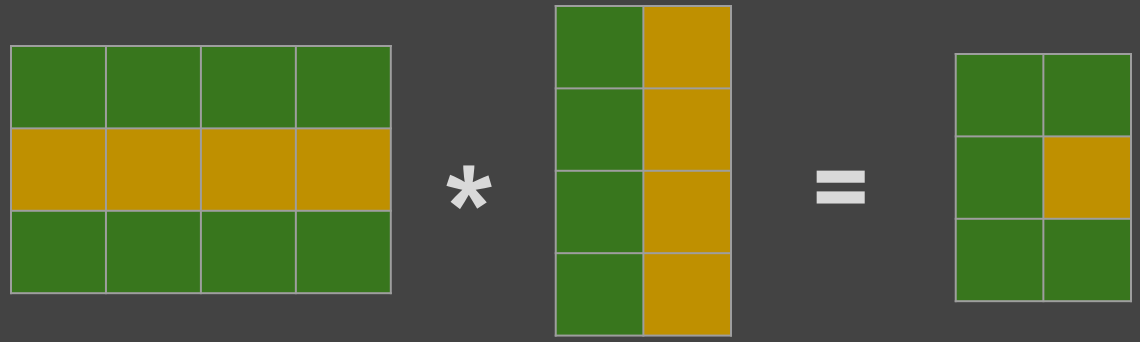
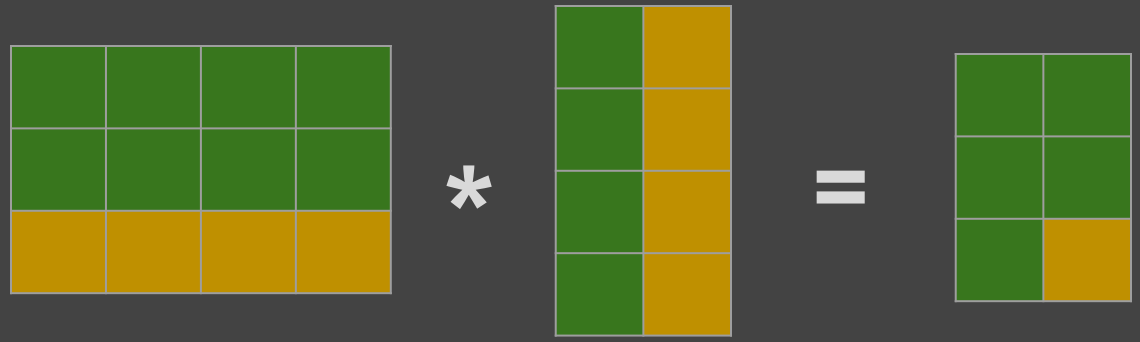

emacs@behrooz-kntu-PC
File Edit Options Buffers Tools C Help
Save
Undo
SSH
/* Standard includes */
/* Our includes */
#include "base.h"
#include "error.h"
#include "convolve.h"
#include "klt_util.h" /* printing */
#define MAX_KERNEL_WIDTH 71
typedef struct {
float data[MAX_KERNEL_WIDTH];
} ConvolutionKernel;
/* Kernels */
Fundamentals of Programming
session 22
2D Arrays, Matrix Multiplication

Multiplying two matrices



Multiplying two matrices



Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8, 7, 6, 5}};

int B[4][2] = {{ 1, 0},
               { 0, 3},
               { 1, 2},
               {-1, 1});

int C[3][2];


int i = 2;
int j = 1;

C[i][j] = ?
```

Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8, 7, 6, 5}};

int B[4][2] = {{ 1, 0},
               { 0, 3},
               { 1, 2},
               {-1, 1}};

int C[3][2];


int i = 2;
int j = 1;

C[i][j] = 8 * 0 + 7 * 3 + 6 * 2 + 5 * 1;
```

Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8, 7, 6, 5}};

int B[4][2] = {{ 1, 0},
               { 0, 3},
               { 1, 2},
               {-1, 1}};

int C[3][2];


int i = 2;
int j = 1;

C[i][j] = A[i][0]*B[0][j] +
          A[i][1]*B[1][j] +
          A[i][2]*B[2][j] +
          A[i][3]*B[3][j];
```

Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8, 7, 6, 5}};

int B[4][2] = {{ 1, 0},
               { 0, 3},
               { 1, 2},
               {-1, 1}};

int C[3][2];


int i = 2;
int j = 1;

C[i][j] = 0;
for (int k = 0; k < 4; k++)
  C[i][j] +=  A[i][k]*B[k][j];
```

Multiplying two matrices

```
int A[3][4] = {{1, 3, 5, 7},
               {2, 4, 6, 8},
               {8,7,6,5}};

int B[4][2] = {{1, 0},
               {0, 3},
               {1, 2},
               {-1, 1}};

int C[3][2];

for (int i = 0; i < 3; i++) {
  for (int j = 0; j < 2; j++) {

    C[i][j] = 0;
    for (int k = 0; k < 4; k++)
      C[i][j] +=  A[i][k]*B[k][j];

  }
}
```

Multiplying two matrices

```c
int main() {
  int A[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {8,7,6,5}};

  int B[4][2] = {{1, 0},
                 {0, 3},
                 {1, 2},
                 {-1, 1}};

  int C[3][2];

  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 2; j++) {

      C[i][j] = 0;
      for (int k = 0; k < 4; k++)
        C[i][j] += A[i][k]*B[k][j];

    }
  }

  print2Darray(3,4, A);
  putchar('\n');

  print2Darray(4,2, B);
  putchar('\n');

  print2Darray(3,2, C);
  putchar('\n');
```

```
nasihatkon@kntu:code$ gcc matrix_mul3.c && ./a.out
   1,   3,   5,   7,
   2,   4,   6,   8,
   8,   7,   6,   5,

   1,   0,
   0,   3,
   1,   2,
  -1,   1,

  -1,  26,
   0,  32,
   9,  38,
```

Matrix multiplication function

```c
#include <stdio.h>

void print2Darray(int m, int n, int a[][n]);
void matrix_mul(int m, int n, int p, int A[m][n], int B[n][p], int C[m][p]);

int main() {

  int A[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {8,7,6,5}};

  int B[4][2] = {{1, 0},
                 {0, 3},
                 {1, 2},
                 {-1, 1}};

  int C[3][2];

  matrix_mul(3,4,2,A,B,C);

  print2Darray(3,4, A);
  putchar('\n');

  print2Darray(4,2, B);
  putchar('\n');

  print2Darray(3,2, C);
  putchar('\n');

  return 0;
}
```

Matrix multiplication function

```
void matrix_mul(int m, int n, int p, int A[m][n], int B[n][p], int C[m][p]) {
  for (int i = 0; i < m; i++) {
    for (int j = 0; j < p; j++) {
      C[i][j] = 0;
      for (int k = 0; k < n; k++)
        C[i][j] +=  A[i][k]*B[k][j];
    }
  }
}
```

```
void matrix_mul(int m, int n, int p, int A[][n], int B[][p], int C[][p]) {
  for (int i = 0; i < m; i++) {
    for (int j = 0; j < p; j++) {
      C[i][j] = 0;
      for (int k = 0; k < n; k++)
        C[i][j] +=  A[i][k]*B[k][j];
    }
  }
}
```

Matrix multiplication function

```c
#include <stdio.h>

void print2Darray(int m, int n, int a[][n]);
void matrix_mul(int m, int n, int p, int A[m][n], int B[n][p], int C[m][p]);

int main() {

  int A[3][4] = {{1, 3, 5, 7},
                 {2, 4, 6, 8},
                 {8,7,6,5}};

  int B[4][2] = {{1, 0},
                 {0, 3},
                 {1, 2},
                 {-1, 1}};

  int C[3][2];

  matrix_mul(3,4,2,A,B,C);

  print2Darray(3,4, A);
  putchar('\n');

  print2Darray(4,2, B);
  putchar('\n');

  print2Darray(3,2, C);
  putchar('\n');

  return 0;
}
```

```
nasihatkon@kntu:code$ gcc matrix_mul4.c && ./a.out
   1,   3,   5,   7,
   2,   4,   6,   8,
   8,   7,   6,   5,

   1,   0,
   0,   3,
   1,   2,
  -1,   1,

  -1,  26,
   0,  32,
   9,  38,
```

Tabular data

	Farzam	Ramin	Bardia	Sobhan
Programming	18	20	15	17
Calculus	12	2	20	18
Farsi	16	16	17	14

Course grades

Tabular data

```
#define FARZAM    0
#define RAMIN     1
#define BARDIA    2
#define SOBHAN    3

#define PROGRAMMING  0
#define CALCULUS     1
#define FARSI        2

#define NO_COURSE  3   // number of courses
#define NO_STUDENT 4   // number of students

double max_course(double grades[][NO_STUDENT], int course);
double max_student(double grades[][NO_STUDENT], int student);
double average_course(double grades[][NO_STUDENT], int course);
double average_student(double grades[][NO_STUDENT], int student);
```

Tabular data

```c
int main() {

  // grades: FARZAM, RAMIN, BARDIA, SOBHAN
  double grades[3][4] = {{18, 20, 15, 17},  // PROGRAMMING
                         {12,  2, 20, 19},  // CALCULUS
                         {16, 16, 17, 14}}; // FARSI


  printf("Max grade of Programming is %.2f\n", max_course(grades, PROGRAMMING));
  printf("Max grade of Sobhan      is %.2f\n", max_student(grades, SOBHAN));
  printf("Average   of Farsi       is %.2f\n", average_course(grades, FARSI));
  printf("Average   of Bardia      is %.2f\n", average_student(grades, BARDIA));


  return  0;
}
```

Compute average/max of a 1D array

```
double avg_array(double array[], int size) {
  double sum = 0;

  for (int i = 0; i < size; i++)
    sum += array[i];

  return sum/size;
}

double max_array(double array[], int size) {
  double max = array[0];

  for (int i = 1; i < size; i++)
    if (array[i] > max)
      max = array[i];

  return max;

}
```

Compute average/max of a course

```
double max_array(double array[], int size);
double avg_array(double array[], int size);
```

```
double max_course(double grades[][NO_STUDENT], int course) {
  return max_array(grades[course], NO_STUDENT);
}
```

```
double average_course(double grades[][NO_STUDENT], int course) {
  return avg_array(grades[course], NO_STUDENT);
}
```

Compute max of a student

```
double max_array(double array[], int size);
double avg_array(double array[], int size);
```

	Farzam	Ramin	Bardia	Sobhan
Programming	18	20	15	17
Calculus	12	2	20	18
Farsi	16	16	17	14

Course grades

Compute max of a student

```
double max_array(double array[], int size);
double avg_array(double array[], int size);
```

```
double max_student(double grades[][NO_STUDENT], int student) {
  double student_grades[NO_COURSE];

  for (int i = 0; i < NO_COURSE; i++)
    student_grades[i] = grades[i][student];

 max_array(student_grades, NO_COURSE);
}
```

Compute average of a student

```
double max_array(double array[], int size);
double avg_array(double array[], int size);
```

```
double average_student(double grades[][NO_STUDENT], int student) {
  double student_grades[NO_COURSE];

  for (int i = 0; i < NO_COURSE; i++)
    student_grades[i] = grades[i][student];

  return avg_array(student_grades, NO_COURSE);
}
```

Running the code

```
int main() {

  // grades: FARZAM, RAMIN, BARDIA, SOBHAN
  double grades[3][4] = {{18, 20, 15, 17},  // PROGRAMMING
                         {12,  2, 20, 19},  // CALCULUS
                         {16, 16, 17, 14}}; // FARSI


  printf("Max grade of Programming is %.2f\n", max_course(grades, PROGRAMMING));
  printf("Max grade of Sobhan      is %.2f\n", max_student(grades, SOBHAN));
  printf("Average   of Farsi       is %.2f\n", average_course(grades, FARSI));
  printf("Average   of Bardia      is %.2f\n", average_student(grades, BARDIA));


  return  0;
}
```

Output

```c
int main() {

  // grades: FARZAM, RAMIN, BARDIA, SOBHAN
  double grades[3][4] = {{18, 20, 15, 17},  // PROGRAMMING
                         {12,  2, 20, 19},  // CALCULUS
                         {16, 16, 17, 14}}; // FARSI


  printf("Max grade of Programming is %.2f\n", max_course(grades, PROGRAMMING));
  printf("Max grade of Sobhan      is %.2f\n", max_student(grades, SOBHAN));
  printf("Average   of Farsi       is %.2f\n", average_course(grades, FARSI));
  printf("Average   of Bardia      is %.2f\n", average_student(grades, BARDIA));


  return  0;
}
```

```
nasihatkon@kntu:code$ gcc tabular1.c && ./a.out
Max grade of Programming is 20.00
Max grade of Sobhan      is 19.00
Average   of Farsi       is 15.75
Average   of Bardia      is 17.33
```

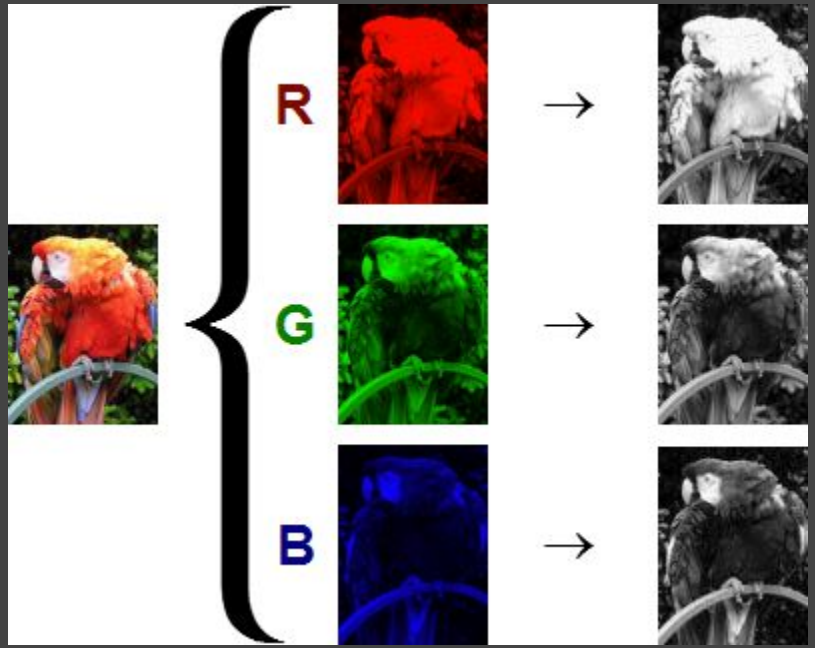
N-dimensional arrays

```
#define FARZAM    0
#define RAMIN     1
#define BARDIA    2
#define SOBHAN    3

#define PROGRAMMING  0
#define CALCULUS     1
#define FARSI        2

#define MIDTERM_EXAM  0
#define FINAL_EXAM    1

#define NO_COURSE  3   // number of courses
#define NO_STUDENT 4   // number of students
#define NO_EXAMS   2   // number of exams


double grades[3][4][2] = {
  { {10,20},{10, 20},{10, 20} },
  { {11,12},{13,14},{15,16} },
  { {20,19},{18,17},{10,9} }
}
```

N-dimensional arrays

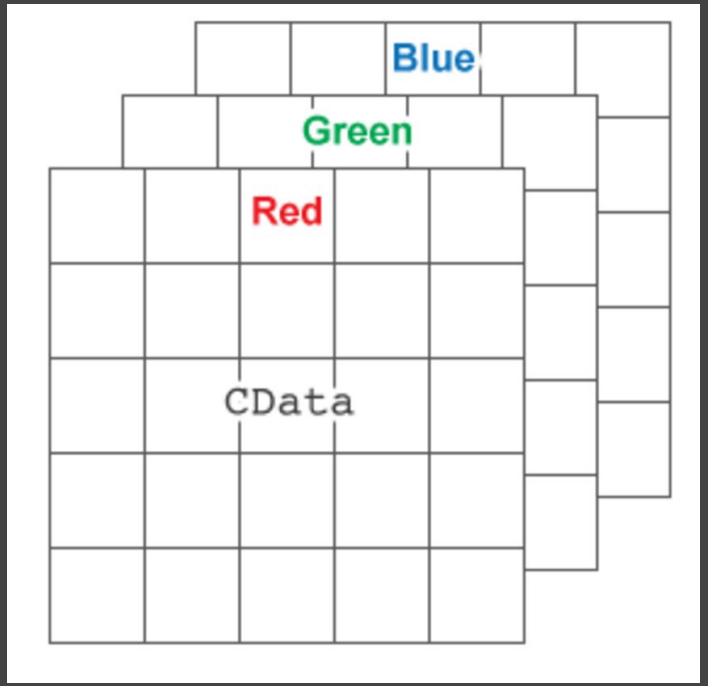
```
double max_course_exam(double grades[][4][2], int course, int exam);
```

Color Images



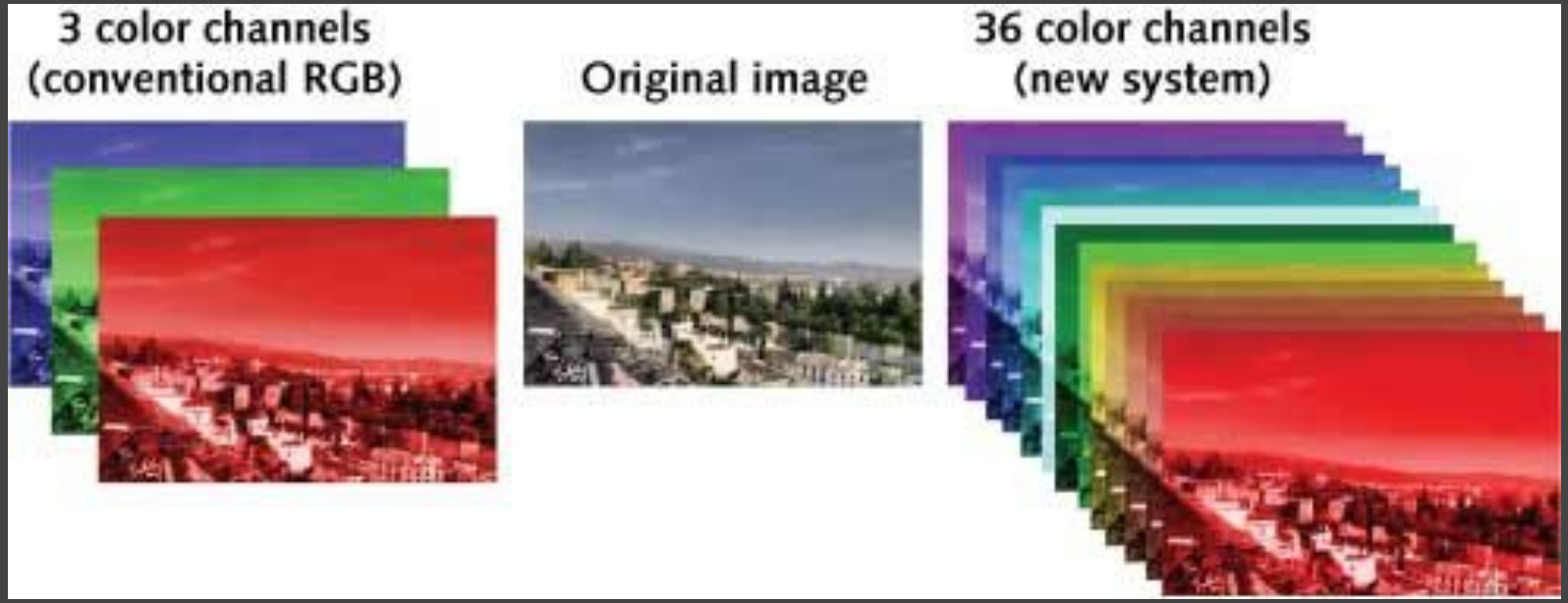
https://commons.wikimedia.org/wiki/File:Beyoglu_4671_tricolor.png#/media/File:RGB_channels_separation.png

Color Images



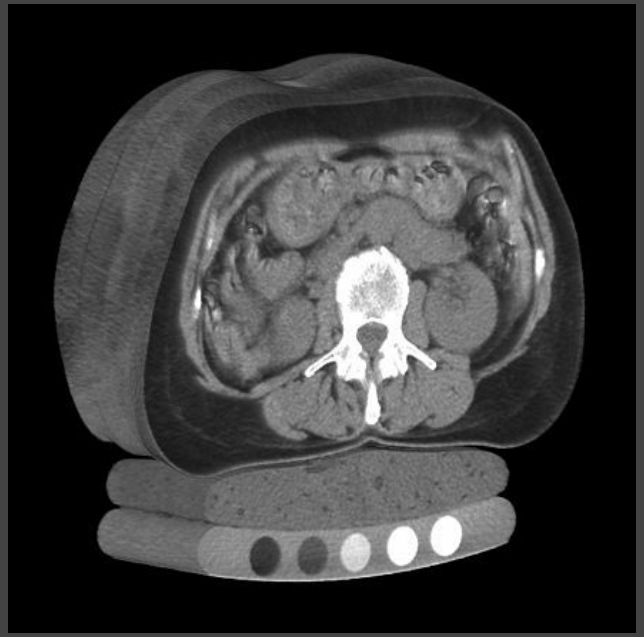
https://www.mathworks.com/help/matlab/ref/image.html?requestedDomain=www.mathworks.com

Multi-spectral images

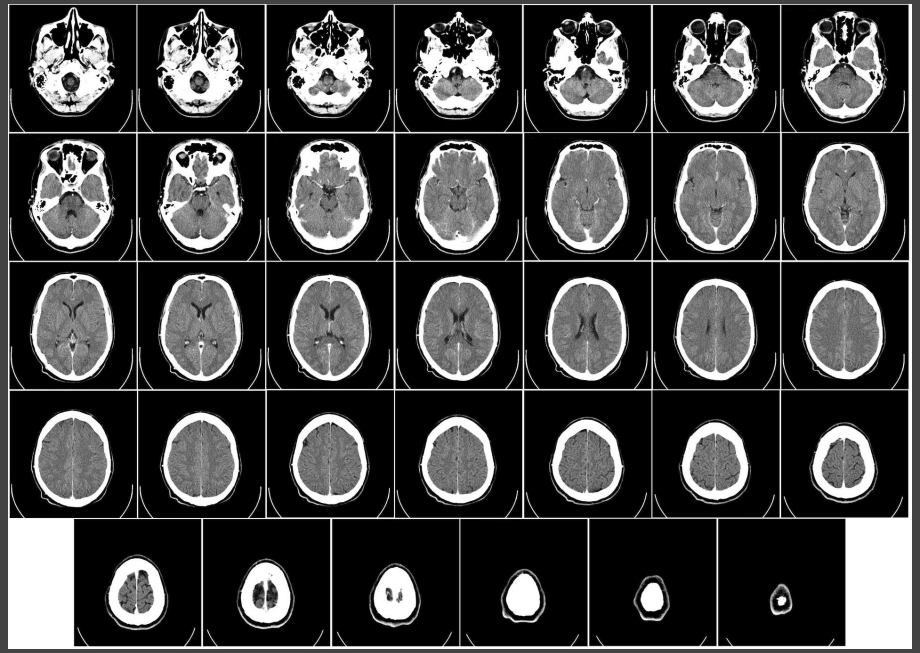


http://www.laserfocusworld.com/articles/print/volume-50/issue-11/world-news/multispectral-imaging-transverse-field-detector-sensor-has-36-color-channels.html

MRI/CT/PET image volume

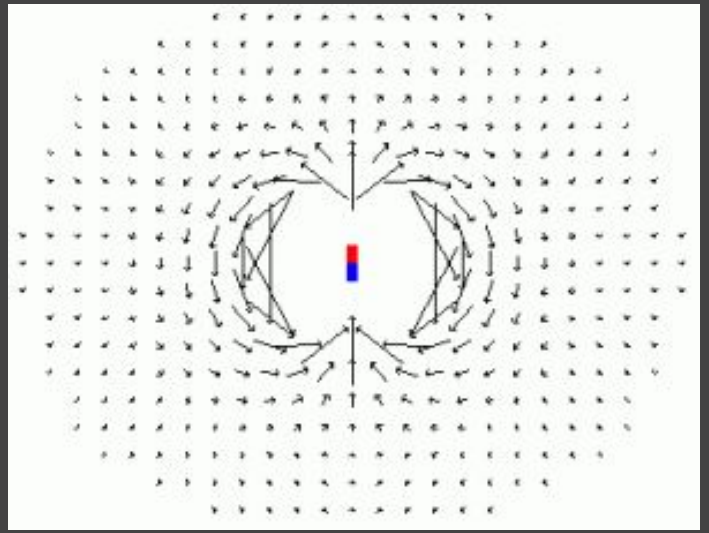


https://en.wikipedia.org/wiki/Volume_rendering

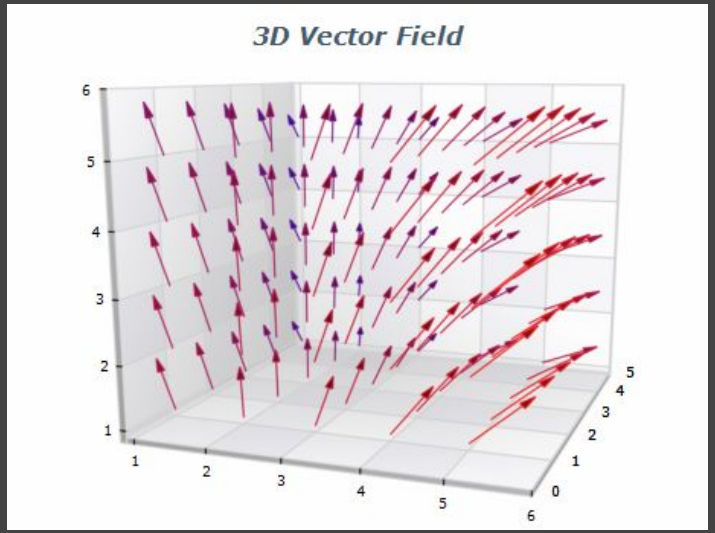


https://en.wikipedia.org/wiki/CT_scan

Vector fields



http://www.evsc.net/home/dipole-spin-system



https://glennmarshall.wordpress.com/2014/10/31/3d-neon-vector-fields/