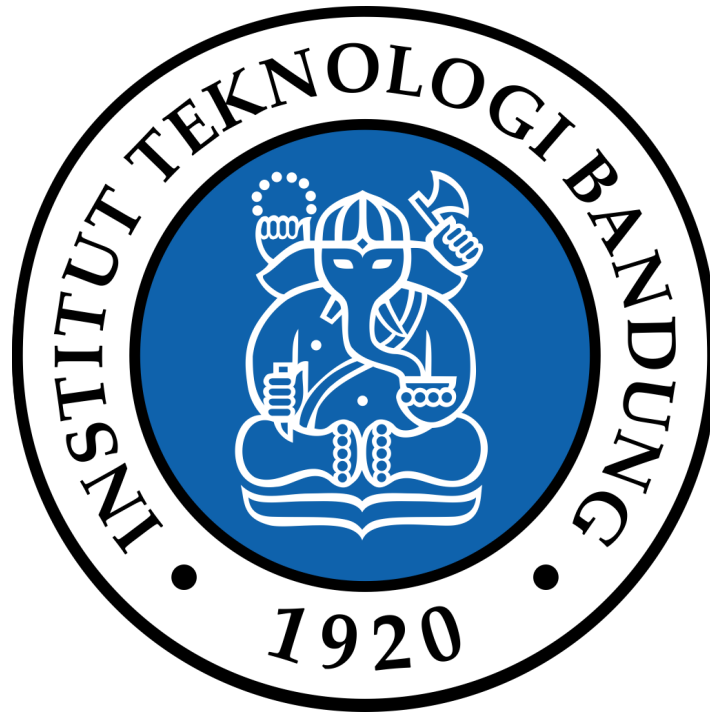


# **IF3260 GRAFIKA KOMPUTER**

## **TUGAS I**

**2D Web Based CAD (Computer Aid Design)**



Disusun oleh:

Yonatan Viody	13518120
Gregorius Jovan Kresnadi	13518135
Stephen Thajeb	13518150

**TEKNIK INFORMATIKA**  
**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**  
**INSTITUT TEKNOLOGI BANDUNG**

# DAFTAR ISI

<b>Deskripsi</b>	<b>3</b>
<b>Hasil</b>	<b>5</b>
<b>Manual</b>	<b>9</b>

## I. Deskripsi

Pada tugas kali ini, kami diminta untuk merancang 2D Computer Aid Design (CAD) berbasis WebGL. Program CAD yang kami rancang menyimpan mode-mode dalam penggambaran sebagai *state*, antara lain:

- a. **Mode Cursor** digunakan untuk melakukan manipulasi seperti pergantian warna atau ukuran pada objek (*entity*) yang telah digambarkan di *canvas*.
- b. **Mode Line** yaitu mode untuk menggambar garis.
- c. **Mode Square** yaitu mode untuk menggambar persegi.
- d. **Mode Polygon** yaitu mode untuk menggambar poligon.

Untuk membuat kode program yang lebih terstruktur dan mudah dibaca, kami menuliskan program dengan pendekatan **Object Oriented**. Secara garis besar, terdapat 5 kelas utama yang kami buat untuk program ini, antara lain:

- a. **Kelas Entity** merepresentasikan *blueprint* untuk sebuah bangun datar yang akan digambar. Kelas ini memiliki atribut berupa:
  - **Vertices** berupa kumpulan titik simpul bangun datar dalam bentuk list/array.
  - **Shape** berupa enumerator untuk garis (*line*), persegi (*square*), dan poligon (*polygon*) yang akan diubah menjadi mode penggambaran yang didukung WebGL.
  - **Color** berupa warna bangun datar dalam format array RGB.
- b. **Kelas EntityBank** merepresentasikan antrian kedatangan **Entity**, yang akan digunakan untuk meload semua bangun datar yang terdefinisi pada *canvas*. Kelas ini memungkinkan semua bangun datar untuk disimpan ke dalam suatu file dan mampu meload file tersebut.
- c. **Kelas MainView** merepresentasikan *canvas* dari semua bangun datar yang sudah digambar pengguna.
- d. **Kelas ShadowView** merepresentasikan *canvas* dari bangun datar yang sedang digambar atau dimanipulasi oleh pengguna.
- e. **Kelas Observer** berfungsi sebagai pengamat yang menerima pesan dari **MainView** dan **ShadowView** dan meneruskannya. Detailnya akan dibahas pada penjelasan di bawah.

Pada program, kami menetapkan beberapa *event listener* terhadap *mouse* (*up*, *down*, and *move*) pada **ShadowView** di mana *event* tersebut akan selalu di-*trigger* pada mode apapun, namun aksi terhadap *event* tersebut berbeda-beda dan akan dijelaskan lebih lanjut di Bab User Manual. Pada saat terjadi *mouse event*, koordinat titik tersebut akan disimpan dalam sebuah *array buffer* dengan index pertama merepresentasikan absis (koordinat x) dan indeks kedua merepresentasikan

ordinat (koordinat y) titik tersebut. Secara umum, yang membedakan respon program kami terhadap mode penggambaran yang berbeda-beda adalah pada ukuran jumlah buffer yang dihasilkan, kecuali untuk **mode cursor**. Sebagai contoh, penggambaran sebuah garis akan menghasilkan buffer berukuran 4 ( $\text{buf} = [x1, y1, x2, y2]$ ), sementara untuk penggambaran sebuah persegi akan menghasilkan buffer berukuran 8 ( $\text{buf} = [x1, y1, x2, y2, x3, y3, x4, y4]$ ).

Untuk mengaplikasikan *user experience* yang baik, kami mencoba menirukan aplikasi kami dengan aplikasi “Paint” di mana dalam penggambaran sebuah bangun datar, kondisi *canvas* akan ikut berubah sesuai dengan *pointer/cursor mouse*, jadi seakan-akan ada *shadow paint* yang tergambar terlebih dahulu pada **ShadowView**, lalu ketika dikonfirmasi oleh user melalui *mouse click*, *shadow paint* tersebut baru akan digambarkan ke **MainView**. Jadi **ShadowView** ini bertugas menampilkan kondisi *canvas* secara *real time* terhadap aksi user. Komunikasi/*passing data* dari **ShadowView** ke **MainView** dilakukan melalui kelas **Observer**. Selain itu, kami juga telah menuliskan beberapa fungsi-fungsi pendukung di file **util.js** yang dapat membantu meningkatkan *reusability* pada program kami, contohnya fungsi untuk membuat persegi dengan 2 titik simpul. Lalu, pada aplikasi kami, kami mengimplementasikan perubahan panjang garis secara *free transform*, yaitu dengan menggerakkan simpul.

## II. Hasil

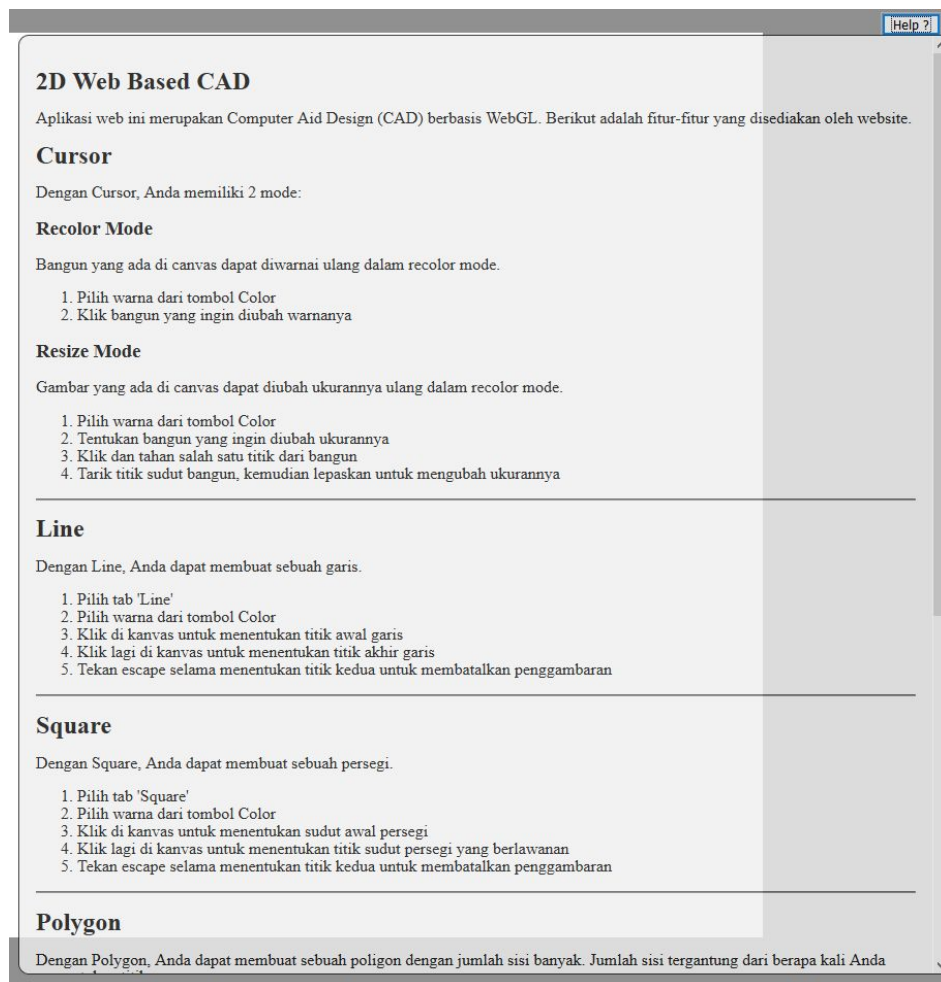
Berikut ditampilkan hasil berupa screenshot tampilan dari program CAD kami.

### 1. Interface program

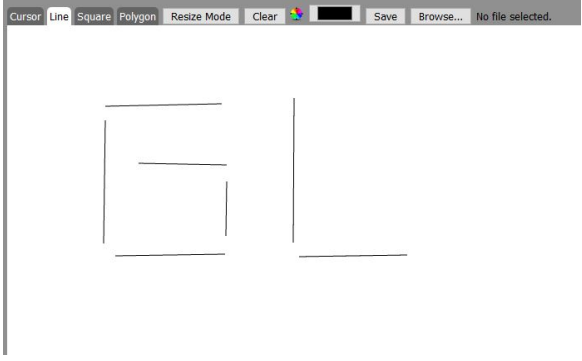
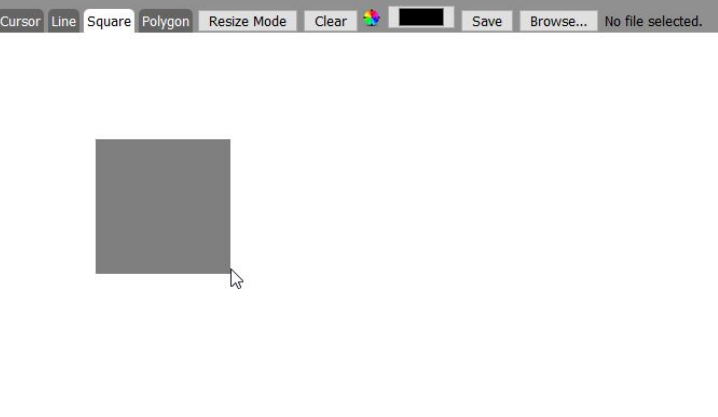

#### a. Halaman Utama



#### b. Kolom user manual



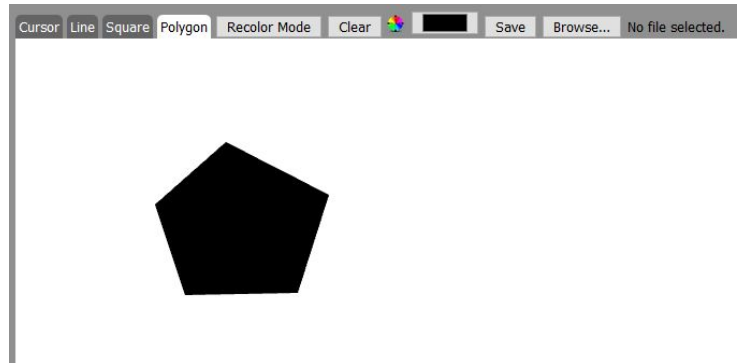
## 2. Penggambaran bangun 2D

Deskripsi	Gambar
<p>1. Penggambaran huruf GL dengan menggunakan garis</p>	
<p>2. Penggambaran Persegi</p> <p>a. Setelah <i>mouse click event</i> pertama di-<i>trigger</i> dan sebelum <i>mouse click event</i> kedua ditrigger. Terlihat bahwa persegi berwarna abu-abu dan ukuran persegi akan berubah sesuai dengan Bergeraknya <i>mouse</i> (<i>mouse move event</i>). Gambar tersebut menunjukkan persegi tersebut telah tergambar pada <i>canvas ShadowView</i>.</p> <p>b. Setelah <i>mouse click event</i> kedua di-<i>trigger</i>, maka entitas yang sebelumnya tergambar di <i>canvas ShadowView</i> akan digambarkan pada <i>canvas MainView</i>. Hal ini diindikasikan oleh warna entitas yang sesuai dengan menu warna yang dipilih.</p>	<p>a.</p>  <p>b.</p> 

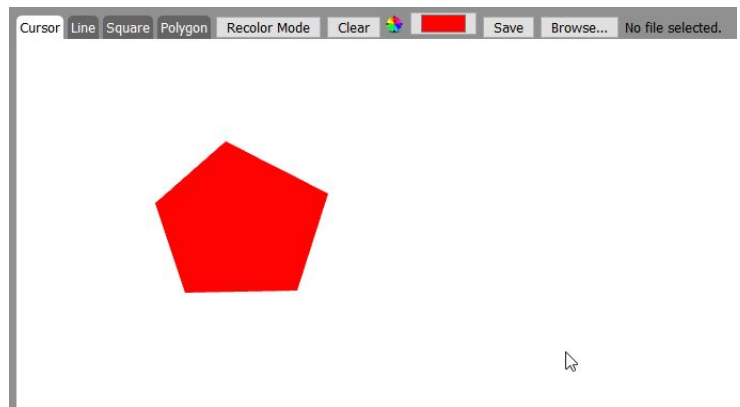
### 3. Penggambaran poligon

- Sebelum dilakukan pergantian warna
- Setelah dilakukan pergantian warna
- Setelah dilakukan pergantian warna dan pergantian ukuran

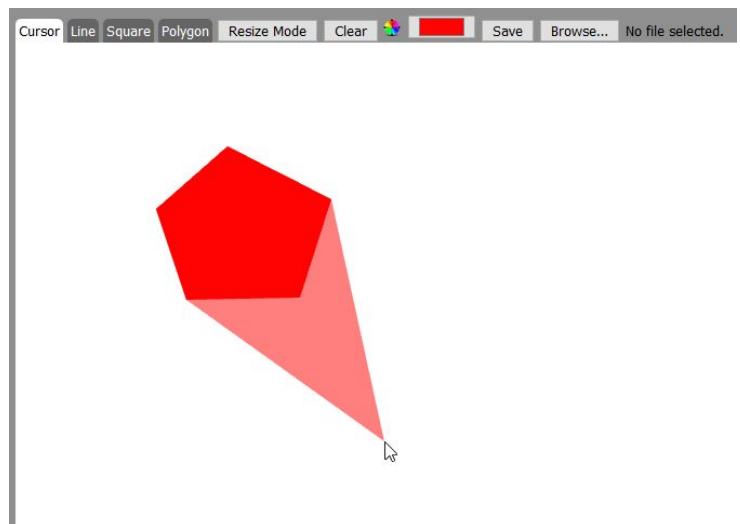
a.

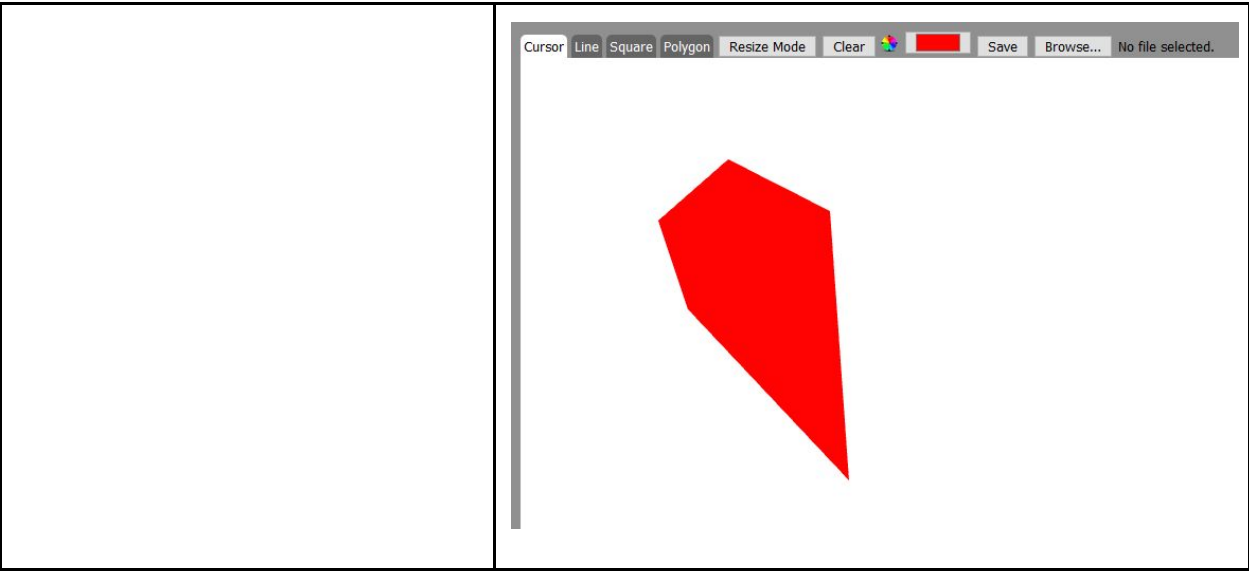


b.



c.

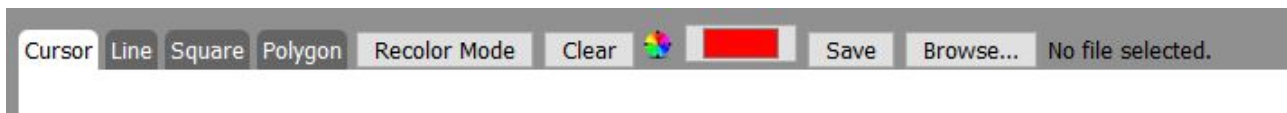






### III. Manual

Untuk lebih memudahkan penjelasan *user manual* dapat melihat potongan foto di bawah ini.



- a. **Button Cursor** akan menggantikan *state* ke **mode cursor** di mana pada *state* ini aksi *resize* dan *recolor* dapat dilakukan. Caranya sebagai berikut:
  - i. men-*toggle* **button Resize / Recolor Mode** sesuai dengan keinginan *edit*.
  - ii. Untuk menggantikan warna cukup pilih warna di **input type color** yang telah disediakan. Kemudian, klik pada entitas yang ingin diganti warnanya.
  - iii. Untuk *resize*, lakukan aksi *mouse click* pada titik simpul dan pindahkan *mouse* (tetap melakukan *mouse click*) sehingga akan terbentuk tampilan entitas di *canvas ShadowView*. Kemudian, untuk finalisasi ukuran entitas tersebut dapat dilakukan aksi *mouse release* (melepas klik).
- b. **Button Line** menggantikan *state* ke **mode line** untuk menggambar sebuah garis, diawali dengan peneklikan sebuah koordinat pada *canvas*, lalu panjang dan arah garis dengan titik asal tersebut dapat disesuaikan dengan pergerakan *mouse* dan difinalisasi dengan peneklikan *mouse*.
- c. **Button Square** adalah mode untuk menggantikan *state* ke **mode square** untuk menggambar persegi. Aksi gambar diawali dengan peneklikan koordinat tertentu pada *canvas*, koordinat tersebut akan menjadi salah satu pojok persegi (penentuan letak koordinat tersebut sebagai pojok kanan atas/bawah, kiri bawah/atas bergantung pada titik kedua yang dipilih), lalu ukuran persegi dapat disesuaikan dengan *mouse move* dan difinalisasi dengan *mouse click* pada koordinat titik berikutnya.
- d. **Button Polygon** akan menggantikan *state* ke **mode polygon** untuk menggambar *polygon*. Aksi awal penggambaran sama seperti penggambaran persegi, pengguna dapat menambahkan banyak titik, dan jumlah titik yang di-*trigger* dari *mouse click* akan menentukan segi poligon tersebut. Untuk aksi finalisasi penggambaran poligon dilakukan dengan cara peneklikan titik awal (daerah di sekitar titik awal). Sebagai contoh untuk menggambar segitiga, seorang pengguna perlu mengklik *mouse* di 4 buah titik, dengan titik ke-4 berada di sekitar titik pertama.
- e. Button kelima dari kiri adalah button untuk **Edit Mode**, peneklikan button tersebut akan mentoggle dari “Resize Mode” menjadi “Recolor Mode” atau sebaliknya. Hal ini berguna untuk menentukan aksi dari **mode cursor** yang akan diterapkan dari interaksi pengguna, yaitu

apakah untuk menggeser simpul dan mengatur ukuran (*resize*) atau melakukan perubahan warna objek sesuai warna yang terpilih pada input (*recolor*).

- f. **Button Clear** berguna untuk menghapus semua entitas yang ada pada *canvas*.
- g. **Input type color** adalah input untuk menerima warna yang akan digunakan untuk gambar entitas di *state line/square/polygon*.
- h. **Button Save** digunakan untuk menyimpan objek-objek di *canvas* ke dalam sebuah file dengan format JSON dengan nama **model.json**.
- i. **Button Choose File** digunakan untuk memload objek-objek yang disimpan pada file JSON sebelumnya dan ditampilkan ke dalam *canvas*.
- j. **Key ESC** digunakan untuk melakukan *cancel* sebuah aksi yang sedang dilakukan pada *canvas* (kecuali *recolor*).