

Поляра на Крива на Безие

Проектът представлява уеб приложение, разработено с *HTML5*, *CSS3* и *JavaScript* (с *WebGL*), за интерактивно създаване и манипулиране на Крива на Безие и нейната първа поляра. Чрез алгоритъма на *de Casteljau* се пресмятат крайната точка на кривата и междинните резултати (полярите), а потребителският интерфейс позволява добавяне, премахване и преместване на контролни точки, както и динамично регулиране на параметъра на интерполация.

Проект разработен от *Теодор Светославов Костадинов*, 4MI0600097 специалност *Софтуерно инженерство*, Факултет по Математика и Информатика, Софийски Университет "Св. Климент Охридски".

1. Задача

Целта на проекта е:

- **Интерактивно изчертаване на Крива на Безие:**
Чрез добавяне на контролни точки върху платното се дефинира формата на кривата.
- **Динамична промяна на интерполацията:**
Параметърът t контролира степента на интерполация, като се пресмятат и визуализират междинните точки.
- **Визуализация на първата поляра:**
Първата итерация от алгоритъма на *de Casteljau* се визуализира отделно, за да се демонстрира процесът на линейна интерполация.
- **Интерактивни операции:**
Потребителят може да добавя, премахва и мести контролни точки чрез мишка и клавиатура, както и да регулира визуалните настройки (размер на точките, стойност на t и т.н.).

2. Техническо описание на функциите

2.1. Инициализация и WebGL конфигурация

Извлича се HTML `<canvas>` елементът по ID (`webgl-canvas`), след което се опитва да се получи контекстът за WebGL. Ако WebGL не е наличен, се използва алтернативният контекст `experimental-webgl`. При неуспех се извежда предупреждение. За визуализация се използват функциите `compileShader(gl, type, source)` и `initShaderProgram(gl, vertexShader, fragmentShader)`.

2.2. Алгоритъмът на *de Casteljau*

Алгоритъмът на *de Casteljau* пресмята точката по Кривата на Безие за дадена стойност на t чрез итеративна линейна интерполация между контролни точки.

- **Математическа дефиниция:**

За $n + 1$ контролни точки P_0, P_1, \dots, P_n се използва следната рекурсивна формула:

$$P_i^{(r)} = (1 - t) \cdot P_i^{(r-1)} + t \cdot P_{i+1}^{(r-1)}, \quad \text{за } i = 0, 1, \dots, n - r,$$

като при $r = 0$ се приема, че $P_i^{(0)} = P_i$. След n итерации се получава крайният резултат:

$$B(t) = P_0^{(n)}.$$

- **Функция `deCasteljau(points, t)`:**

- **Описание:** Изчислява точката $B(t)$ по Крива на Безиета за даден масив от контролни точки и стойност на t .
- **Действие:**
 1. Копира входния масив от точки в локален масив `tmpPoints`.
 2. Чрез вложени цикли итеративно пресмята нови междинни точки чрез формулата за *de Casteljau*.
 3. Връща крайната точка, намираща се в `tmpPoints[0]`.

2.3. Рендериращи функции

- **Функция `setupBuffer(vertices)`:**

- **Описание:** Създава и настройва WebGL буфер за зададен вектор от върхови координати.
- **Действие:**
 1. Създава буфер, свързва го към `ARRAY_BUFFER`.
 2. Зарежда данните в буфера.
 3. Настройва атрибута за позиция и връща буфера.

- **Функция `renderPoints(points, color, size)`:**

- **Описание:** Изчертава точки върху платното.
- **Действие:**
 1. Преобразува масива от точки в един вектор от координати.
 2. Използва `setupBuffer` за създаване на буфер.
 3. Задава цвета чрез променливата `uColor`.
 4. Задава размера на точките чрез `uPointSize` и чертае точките с `gl.POINTS`.

- **Функция `renderStraightLines(points, color)`:**

- **Описание:** Изчертава отсечките между дадени точки.
- **Действие:**
 1. Проверява дали има поне 2 точки.
 2. Преобразува точките във вектор от координати.
 3. Използва `setupBuffer` за създаване на буфер.
 4. Задава цвета и чертае линията с `gl.LINE_STRIP`.

- Функция `renderCurve(controlPoints, color, mode):`

- **Описание:** Изчислява и визуализира крива, базирана на даден набор от контролни точки, чрез итеративното пресмятане с алгоритъма на *de Casteljau*.
- **Действие:**
 1. За всяка стойност на t от 0 до 1 (стъпка 0.01) се изчислява точка по кривата.
 2. Полученият вектор от върхови координати се задава чрез `setupBuffer`.
 3. Задава се цвят и се чертае крива линия.

- Функция `computeIntermediatePoints(points, t):`

- **Описание:** Изчислява междинните точки за първата итерация от алгоритъма, използвайки линейна интерполация между всяка двойка съседни контролни точки.
- **Действие:** За всяка двойка $[P_i, P_{i+1}]$ се пресмята:

$$P_i^{(1)} = \left((1 - t) \cdot P_i[0] + t \cdot P_{i+1}[0], (1 - t) \cdot P_i[1] + t \cdot P_{i+1}[1] \right)$$

и се добавя към нов масив, който се връща.

- Функция `render():`

- **Описание:** Основната функция за обновяване на платното, която обединява всички визуализиращи операции.
- **Действие:**
 1. Изчиства платното и задава бял фон.
 2. Настройва WebGL програмата.
 3. Изчертава контролните точки и полигонът, който ги свързва.
 4. Изчертава крайната Крива на Безие (ако има поне 3 точки).
 5. Ако е активирана опцията, изчертава междинните точки и линиите между тях.
 6. Ако е активирана опцията, изчертава първата поляра.

2.4 Обработка на потребителски събития и управление на състоянието

- **Промяна на размера на точките:**

- Функциите `increasePointSize()` и `decreasePointSize()` увеличават или намаляват визуалния размер на точките. След промяната се извиква функцията `render()` за актуализиране на платното.

- **Добавяне, премахване и преместване на точки:**

- **Ляв бутон:** При нормален клик се добавя нова контролна точка (новата точка се добавя в масива `points`).
- **Shift + клик:** При задържане на Shift, ако точката е избрана (определя се чрез функцията `findClickedPointIndex()`), тя се премахва от масива, след което се извиква `render()`.
- **Ctrl + влачене:** При задържане на Ctrl се мести избраната точка – по време на `mousemove` позицията ѝ се обновява в масива `points` и се извиква `render()`.

- **Промяна на параметъра t :**

- Стойността на t се регулира чрез плъзгача (`t-slider`) или клавишите [и]. При всяка промяна се обновява дисплеят с текущата стойност и се извиква `render()`. Функциите `increaseSliderValue()` и `decreaseSliderValue()` отговарят за това.

- **Обработка на клавишни комбинации:**

- Клавишите + и - са свързани с функциите `increasePointSize()` и `decreasePointSize()`.
- Клавишът z премахва последната добавена точка чрез функцията `removeLastPoint()`.
- Клавишът i превключва визуализацията на междинните точки чрез функцията `toggleIntermediatePoints()`.
- Клавишът c изтрива всички контролни точки чрез функцията `clearAllPoints()`.
- Клавишът r нулира настройките към стойностите по подразбиране чрез функцията `resetToDefaults()`.
- Клавишите [и] регулират параметъра t чрез извикване на функциите `decreaseSliderValue()` и `increaseSliderValue()`.
- Клавишът p превключва визуализацията на първата поляра чрез функцията `toggleFirstPolar()`.

3. Математическо обяснение на първата поляра

Първата поляра се получава чрез прилагане на линейна интерполация между всяка двойка съседни контролни точки, използвайки същата формула като в алгоритъма на *de Casteljau*, но само за първата итерация:

$$P_i^{(1)} = (1 - t) \cdot P_i + t \cdot P_{i+1}, \quad i = 0, 1, \dots, n - 1.$$

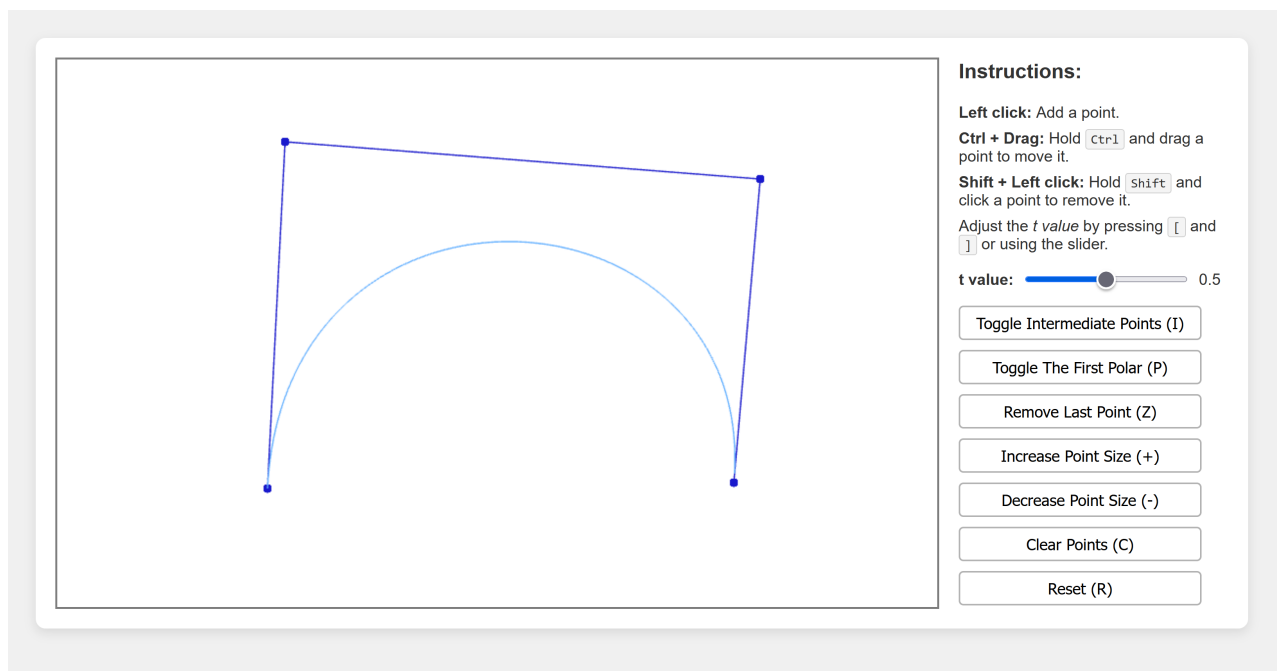
След това, чрез прилагане на алгоритъма на *de Casteljau* върху получения набор от междинни точки, се пресмята крайната точка $B(t)$ на полярната крива. Тази визуализация подпомага разбирането на последователния процес на интерполация, който води до определяне на крайната точка на оригиналната Крива на Безие.

4. Потребителски интерфейс

4.1. Мишка

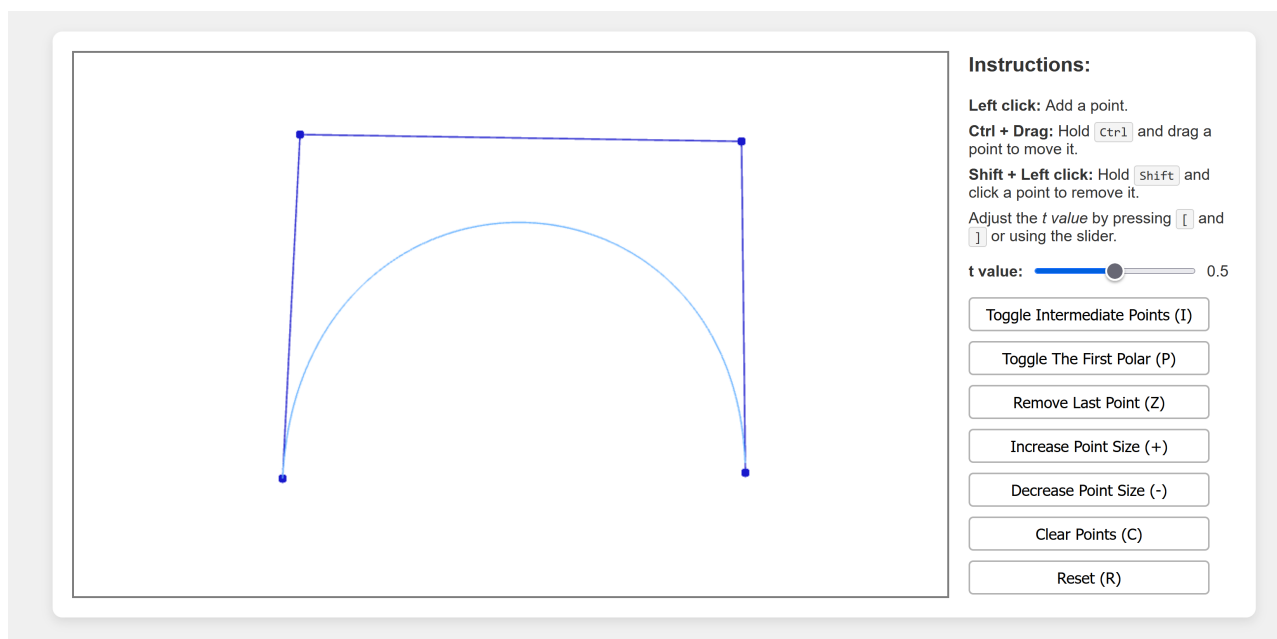
- **Ляв бутон:**

Добавяне на нова контролна точка върху платното.



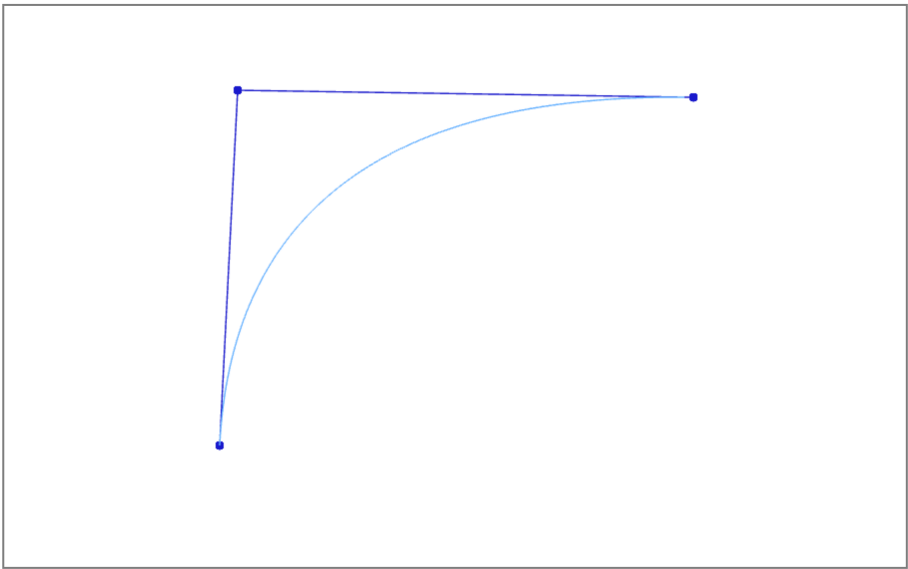
- **Ctrl + влачене:**

Преместване на избрана точка – позицията ѝ се обновява в реално време, като кривата се пресмята отново.



- **Shift + клик:**

Премахване на конкретна точка от контролния масив.



Instructions:

Left click: Add a point.

Ctrl + Drag: Hold `Ctrl` and drag a point to move it.

Shift + Left click: Hold `Shift` and click a point to remove it.

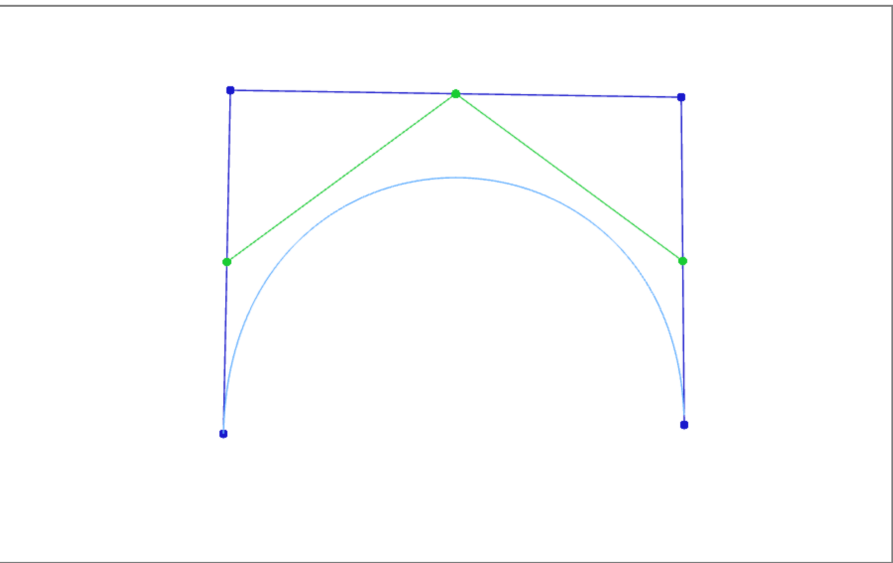
Adjust the t value by pressing `[` and `]` or using the slider.

t value: 0.5

4.2. Плъзгач

- **Стойност на t :**

Регулира се чрез плъзгача или клавишите `[` и `]`. Тази стойност определя интерполационния коефициент, използван за изчисляване на междинните резултати.



Instructions:

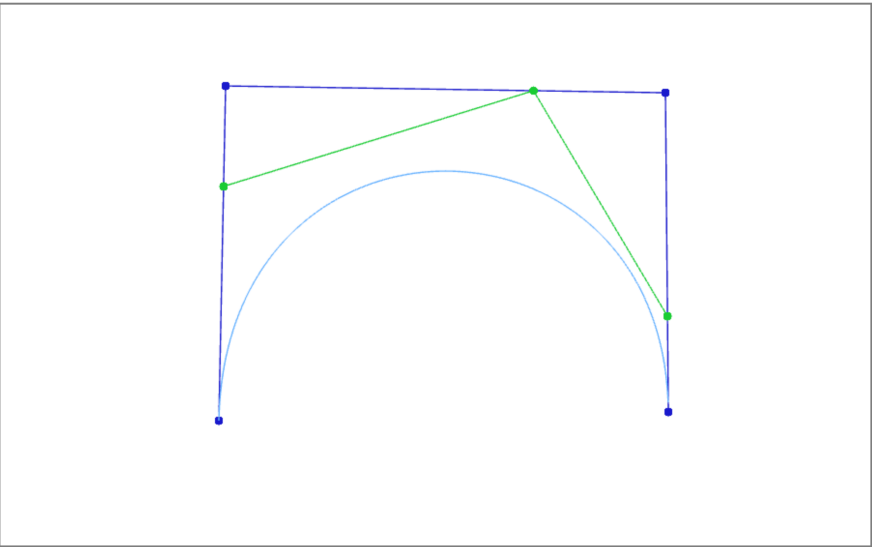
Left click: Add a point.

Ctrl + Drag: Hold `Ctrl` and drag a point to move it.

Shift + Left click: Hold `Shift` and click a point to remove it.

Adjust the t value by pressing `[` and `]` or using the slider.

t value: 0.5



Instructions:

Left click: Add a point.

Ctrl + Drag: Hold `Ctrl` and drag a point to move it.

Shift + Left click: Hold `Shift` and click a point to remove it.

Adjust the t value by pressing `[` and `]` or using the slider.

t value: 0.7

Toggle Intermediate Points (I)

Toggle The First Polar (P)

Remove Last Point (Z)

Increase Point Size (+)

Decrease Point Size (-)

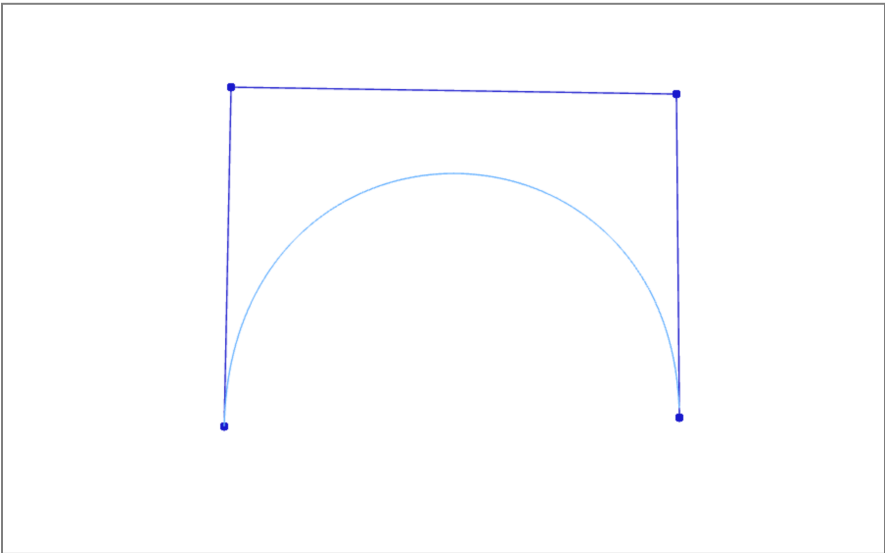
Clear Points (C)

Reset (R)

4.3. Бутони

- **Toggle Intermediate Points (I):**

Превключва визуализацията на междинните точки, пресметнати чрез алгоритма *de Casteljau*.



Instructions:

Left click: Add a point.

Ctrl + Drag: Hold `Ctrl` and drag a point to move it.

Shift + Left click: Hold `Shift` and click a point to remove it.

Adjust the t value by pressing `[` and `]` or using the slider.

t value: 0.7

Toggle Intermediate Points (I)

Toggle The First Polar (P)

Remove Last Point (Z)

Increase Point Size (+)

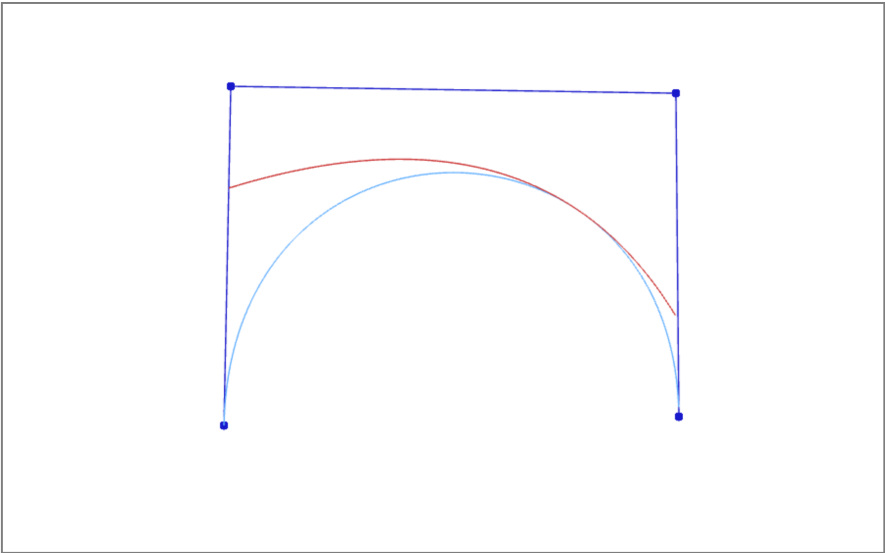
Decrease Point Size (-)

Clear Points (C)

Reset (R)

- **Toggle The First Polar (P):**

Превключва визуализацията на първата поляра (получената от първата итерация).



Instructions:

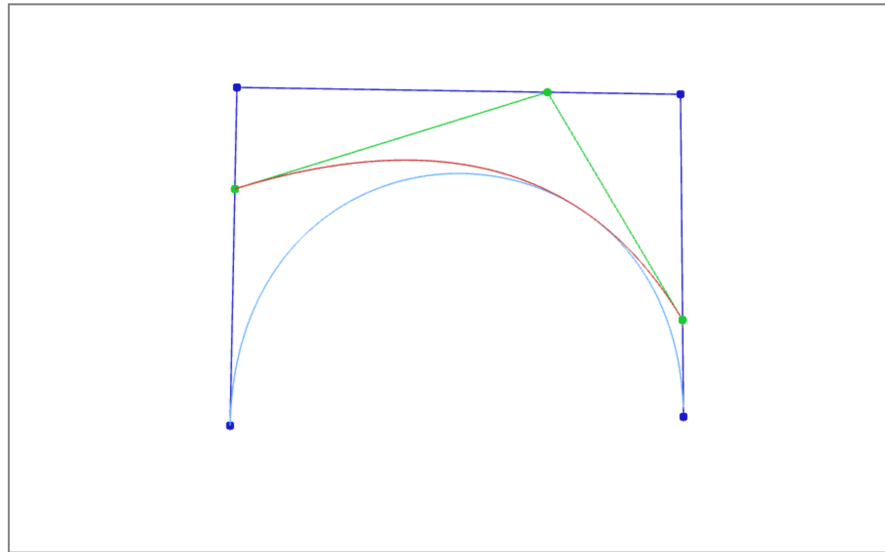
Left click: Add a point.

Ctrl + Drag: Hold **Ctrl** and drag a point to move it.

Shift + Left click: Hold **Shift** and click a point to remove it.

Adjust the *t* value by pressing **t** and **j** or using the slider.

t value: 0.7



Instructions:

Left click: Add a point.

Ctrl + Drag: Hold **Ctrl** and drag a point to move it.

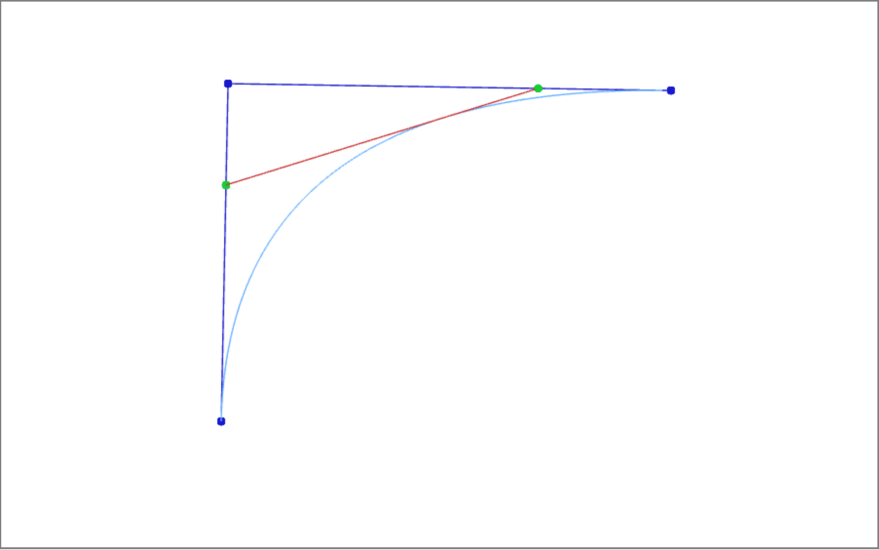
Shift + Left click: Hold **Shift** and click a point to remove it.

Adjust the *t* value by pressing **t** and **j** or using the slider.

t value: 0.7

- **Remove Last Point (Z):**

Последната добавена контролна точка се премахва.



Instructions:

Left click: Add a point.

Ctrl + Drag: Hold `ctrl` and drag a point to move it.

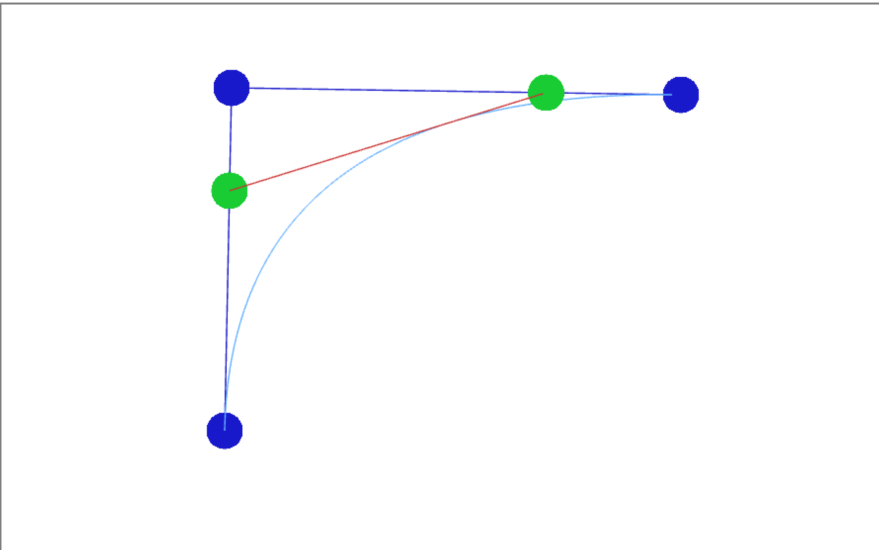
Shift + Left click: Hold `shift` and click a point to remove it.

Adjust the *t value* by pressing `[` and `]` or using the slider.

t value: 0.7

- **Increase Point Size (+) / Decrease Point Size (-):**

Размерът на визуализираните точки се увеличава или намалява. Помага за по-добро управление на позицията на точките чрез другите функции.



Instructions:

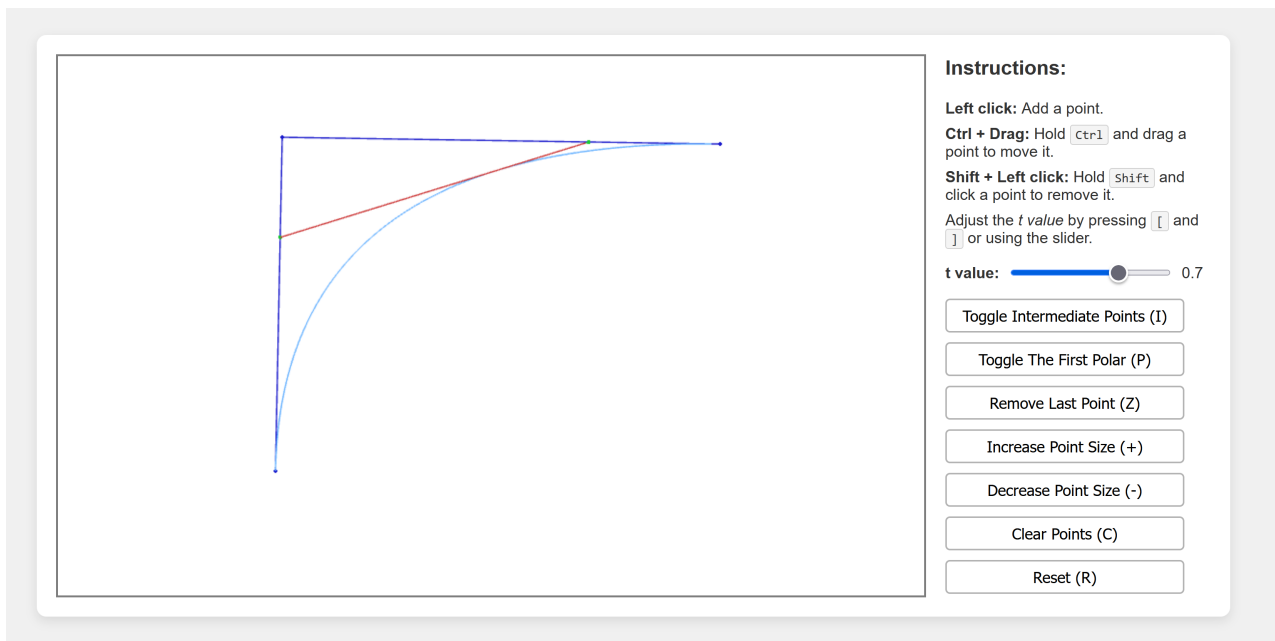
Left click: Add a point.

Ctrl + Drag: Hold `ctrl` and drag a point to move it.

Shift + Left click: Hold `shift` and click a point to remove it.

Adjust the *t value* by pressing `[` and `]` or using the slider.

t value: 0.7



- **Clear Points (C):**

Всички контролни точки се изтриват, като визуализацията се нулира. Запазват се останалите настройки.



- **Reset (R):**

Връща настройките към стойностите по подразбиране (контролни точки, размер на точките и стойност на t).



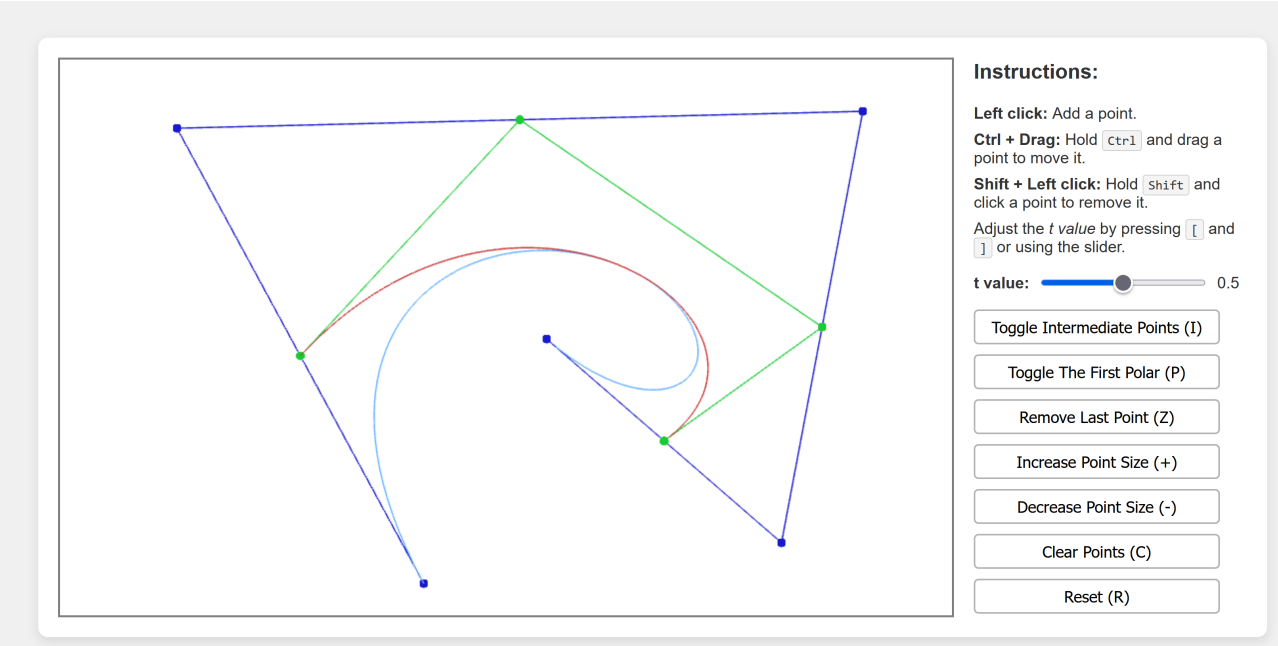
The screenshot shows a software interface with a large empty rectangular canvas on the left. To the right of the canvas is a control panel titled "Instructions:". The panel contains the following text and controls:

- Left click:** Add a point.
- Ctrl + Drag:** Hold `Ctrl` and drag a point to move it.
- Shift + Left click:** Hold `Shift` and click a point to remove it.
- Adjust the t value by pressing `[]` or using the slider.
- t value:** A slider bar with a blue gradient, a black knob, and the value "0.5" on the right.
- Buttons: "Toggle Intermediate Points (I)", "Toggle The First Polar (P)", "Remove Last Point (Z)", "Increase Point Size (+)", "Decrease Point Size (-)", "Clear Points (C)", and a blue "Reset (R)" button.

4.4. Визуални елементи

- **Контролни точки (сини точки):**

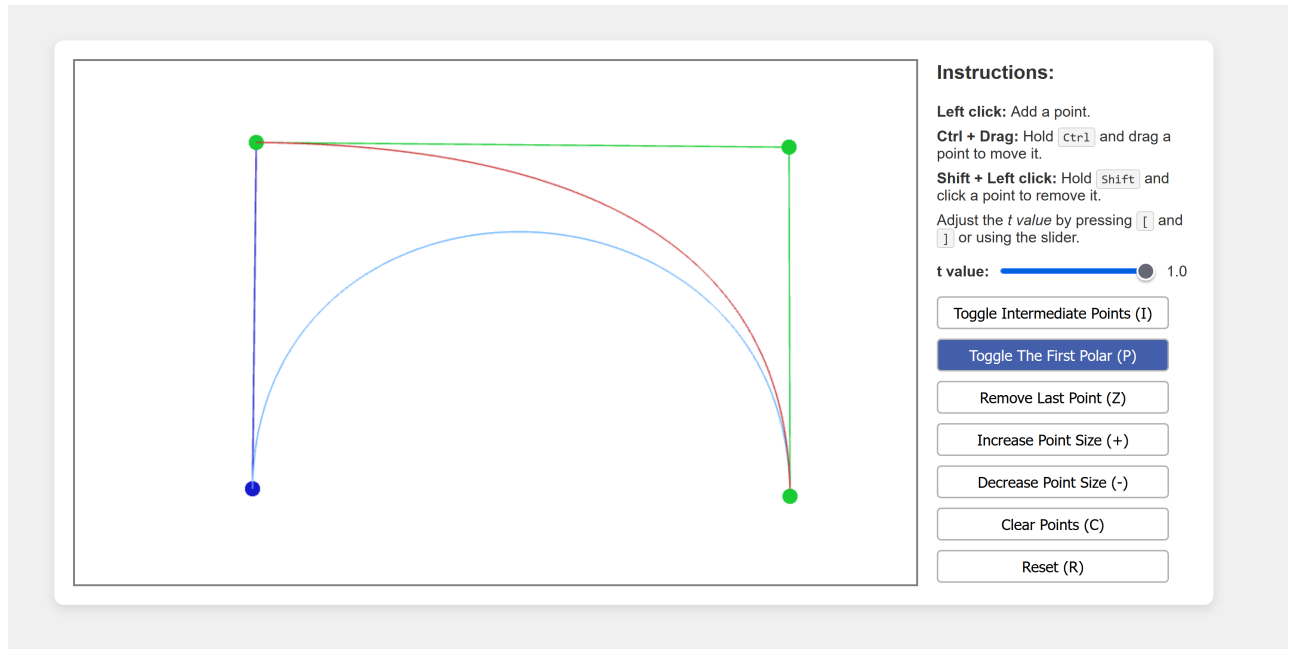
Показват текущите контролни точки, които дефинират формата на кривата.



The screenshot shows the same software interface as before, but the canvas now displays a curve. The curve is a smooth, S-shaped line that starts at a blue point on the left, goes up and right, then loops back down and right to end at a blue point on the right. There are four green control points: two on the left side of the curve and two on the right side. Lines connect these green points to form a quadrilateral that encloses the curve. The control panel on the right is identical to the one in the previous screenshot.

- **Междинни точки (зелени точки):**

Изобразяват резултатите от първата итерация на алгоритъма на *de Casteljau*, илюстрирайки линейната интерполация. Междинните точки след първата итерация могат да съвпадат с контролните при $t = 0$ и $t = 1$.



- **Крива на Безие (светлосиня линия):**

Кривата, пресметната чрез *de Casteljau* върху оригиналните контролни точки.

- **Първа поляра (червена линия):**

Кривата, получена чрез прилагане на *de Casteljau* върху междинните точки (първата итерация).

5. Източници

- Материали от курса "Компютърно Геометрично Моделиране", Факултет по Математика и Информатика, Софийски Университет "Св. Климент Охридски".
- [WebGL Fundamentals](#)
- [WebGL Tutorial Setup](#)