# MAC0468 - Tópicos de Computação Gráfica

César Gasparini Fernandes
Número USP: 10297630

# Construção e Simulação Virtual de Montanhas-Russas Voltadas para Jogos Digitais

Segundo semestre de 2018

# 1 Introdução

## 1.1 Problema Abordado

Uma das atrações mais populares dos parque de diversão é a montanha-russa. Ela se constitui de uma múltipla rede de trilhos, armados em aclives e declives sucessivos, e através dos quais circula, com relativa velocidade, uma espécie de trem (que chamaremos de "carro"), composto de pequenos compartimentos abertos com barras de ferro e bancos nos quais as pessoas se sentam[1].

As montanhas-russas são também exploradas no mundo virtual, fortemente presentes em jogos de construção de parques de diversão e em simuladores de montanha-russa nos óculos de realidade virtual.

Conforme foram passando os anos, os jogos foram melhorando seus construtores de montanha-russa, de seleção de segmentos de trilhos pré-definidos antigamente, passando para seleção das angulaturas e comprimento do segmento de trilho.

## 1.2 Motivações

Desde criança, tinha curiosidade de saber como eram feitas as montanhas-russas nos jogos e, recentemente, com o lançamento do jogo Planet Coaster®, quis saber como era feito para "entortar" o trilho.

Com isso, gostaria de aprender a técnica de extrusão, lidar com curvas de Bézier, aplicar álgebra e cálculo e poder manipular um modelo diretamente pelos seus vértices.

## 1.3 Objetivos

Os objetivos desse projeto foram desenvolver uma ferramenta para construir uma montanha-russa virtual que fosse de fácil uso e simular um carro nela.

# 2 Metodologia

Esse capitulo foi dividido em 2, a primeira parte se refere a como foi desenvolvido a construção da montanha-russa, a segunda parte de como são simulados os carros.

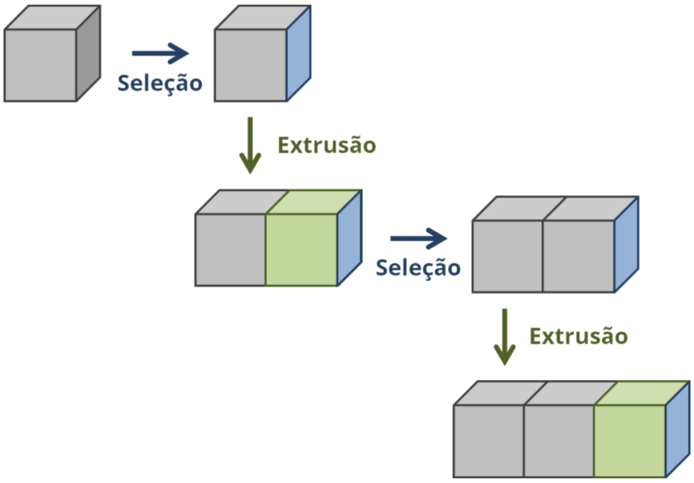
## 2.1 Construção

Para começar a construir a montanha-russa, temos um conjunto de pontos de uma curva (que forma uma circunferência) que descreve um segmento infinitesimal do trilho. Vamos ver a seguir como transformar essa curva no trilho e o conjunto de trilhos na montanha-russa.

### 2.1.1 Extrusão

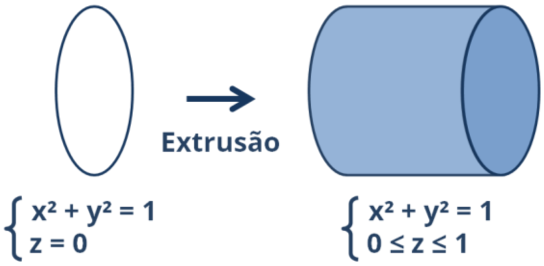
A técnica de extrusão Gera um sólido ou uma superfície 3D ao estender uma curva ou uma superfície 2D ao longo de uma curva. Ela é uma técnica elementar no processo de modelagem, tanto que está presente nos softwares de modelagem. Com ela é possível transformar um cubo em uma espaçonave por exemplo.

A seguir, vamos ver como essa técnica funciona.



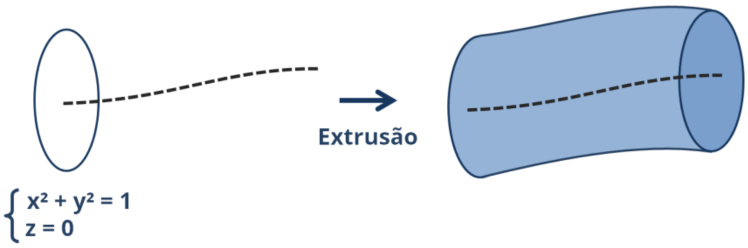
Esquema 2.1.1.1: aplicação da técnica de extrusão a partir de uma superfície

O esquema a cima mostra como a técnica de extrusão é feita a partir de uma superfície. Primeiro é selecionado uma superfície (em azul - face apontando para a direita) e então ela é extrusada, criando faces em verde - as novas faces em comparação com o modelo anterior.



Esquema 2.1.1.2: aplicação da técnica de extrusão a partir de uma curva

A técnica de extrusão aplicada a uma curva gera uma face que acompanha essa curva, como no esquema a cima.



Esquema 2.1.1.3: aplicação da técnica de extrusão seguindo uma curva (em tracejado)

Uma propriedade importante da técnica de extrusão é que ela permite que a curva que está sendo extrosada possa seguir uma outra curva como no esquema a cima. Essa outra curva pode ser uma curva de Bézier que vamos ver a seguir.

### 2.1.2 Curvas de Bézier

As curvas de Bézier são polinômios de grau n descritos por n+1 pontos e definidos pela seguinte equação:

$$B(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i P_i$$

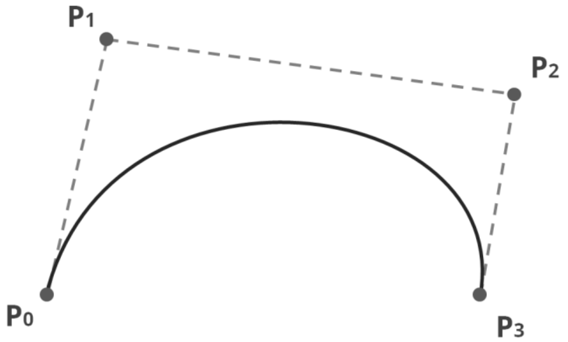
Onde $B(t)$ é a função da curva, $t$ é uma variável, $n$ é o número de pontos menos 1, e $P_i$ é o i-ésimo ponto.

Embora a curva de Bézier possa ser de grau $n$ (ou seja, $n+1$ pontos), para a montanha-russa foi utilizado de grau 3 (ou seja, 4 pontos).

Sua fórmula é definida do seguinte modo:

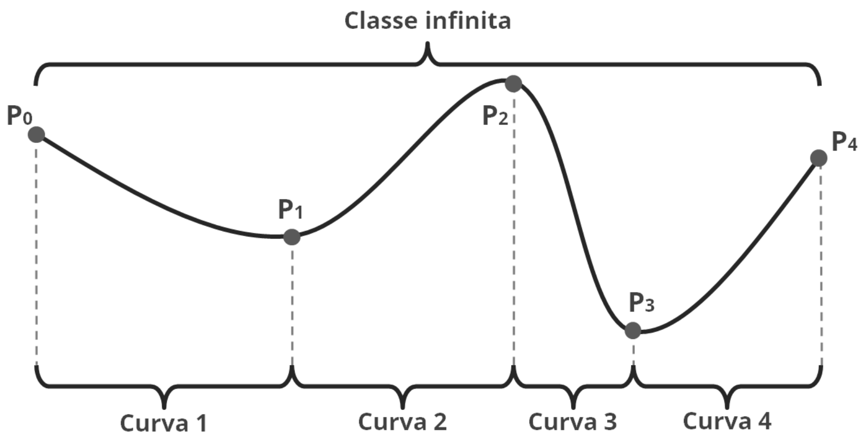
$$B(t) = (1-t)^3 P_0 + 3(1-t)^2 t^1 P_1 + 3(1-t)^1 t^2 P_2 + t^3 P_3$$

E seu gráfico é dado do seguinte modo:




Esquema 2.1.2.1: gráfico de uma curva de Bézier cúbica

Uma de suas propriedades interessantes é a possibilidade de controlar as derivadas em B(0) e B(1) utilizando P1 e P2, ela torna possível com que composições de Bézier sejam de classe infinita (infinitamente derivável) se os pontos e tangentes finais e iniciais no intervalo [0,1] coincidirem.

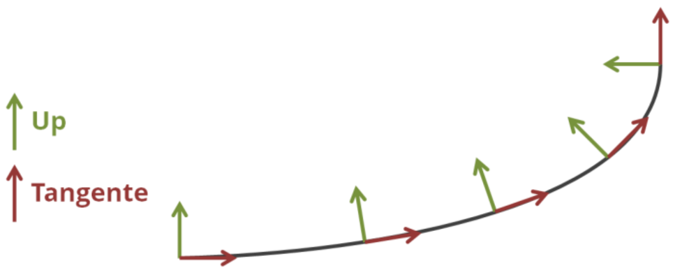



Esquema 2.1.2.2: gráfico mostrando que composições de curvas de Bézier são de classe infinita.

### 2.1.3 Transformação de base

Agora que sabemos como replicar os pontos  da curva a ser extrusada ao longo de uma curva de Bézier, temos que saber como fazer para a orientação deles seguir a orientação da curva de Bézier.

Um problema, entretanto, é que uma curva não tem orientação no $R^3$. O que devemos fazer, portanto, é atribuir uma orientação inicial a curva e transformá-la ao longo da curva.




Esquema 2.1.3.1: esquema mostrando orientação ao longo da curva.

Para isso, vamos considerar que a tangente $B'(0)$ seja $(1, 0, 0)$. Então atribuamos o vetor $up = (0, 1, 0)$ à orientação. Com esses dois vetores perpendiculares, podemos deduzir o terceiro da base, que chamaremos de cross, pelo produto vetorial entre a tangente e up.

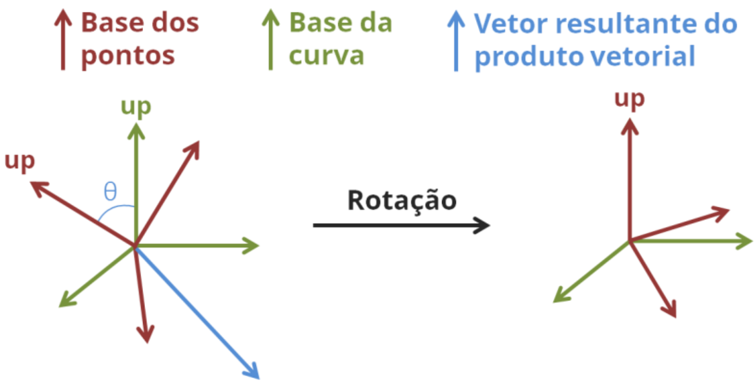
Temos então uma base ortonormal que representa a orientação ao longo da curva. Agora devemos orientar os pontos da curva a ser extrusada. Uma solução para esse problema é a transformação de base.

A partir de agora, os pontos da curva a ser extrusada serão chamados apenas de pontos e a curva de Bézier será chamada apenas de curva.

Inicialmente começamos com a seguinte configuração de bases (orientações):

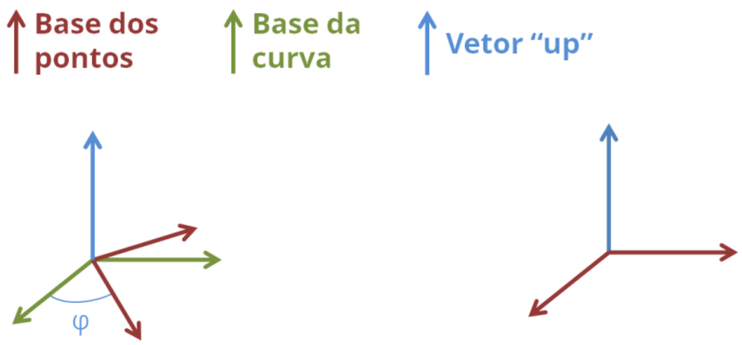

Esquema 2.1.3.2: esquema mostrando configuração inicial das bases.

Para transformar as bases, basta fazer duas rotações. Primeiro rotacione ao redor do vetor resultante do produto vetorial entre os vetores up das duas bases para que eles tenham mesma direção e sentido




Esquema 2.1.3.3: esquema mostrando a primeira rotação da base dos pontos.

Em seguida, rotacione ao redor do vetor up para que as bases fiquem iguais.



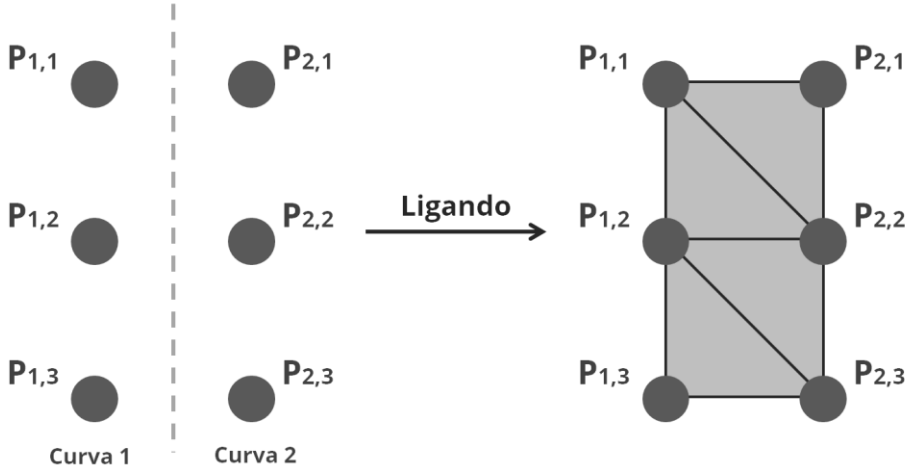

Esquema 2.1.3.4: esquema mostrando a segunda rotação da base dos pontos.

Depois dessas duas rotações, temos uma transformação de base da base dos pontos para a base da curva.

Logo temos os pontos da curva a ser extrusada posicionados e orientados ao longo da curva de Bézier.

### 2.1.4 Ligando os pontos

Para a criação de faces no modelo extrusado, basta "ligar" os pontos da maneira como está no esquema abaixo:



Esquema 2.1.4: criação de faces no modelo extrusado, fazendo triangulações.

### 2.1.5 Rotação

O jogador possui 4 graus de liberdade para construir sua montanha-russa:

- Comprimento do segmento;
- Elevação;
- Rotação;
- Inclinação.

Sendo que a elevação, rotação e inclinação pertencem a [-90°, 90°] ∈ $Z$ e o comprimento pertence a [1, 20] ∈ $N$.

A ordem que o programa avalia os atributos dados pelo usuário é:

**Elevação** –> **Rotação** -> **Inclinação** –> **Comprimento**

Para seguir os parâmetros dados pelo jogador, o ultimo segmento construído "entra" perpendicularmente em uma esfera imaginária  para calcular a forma do próximo segmento seguindo os parâmetros definidos pelo usuário.

O raio direcional serve para sabermos onde termina o segmento de trilho e qual sua tangente final.

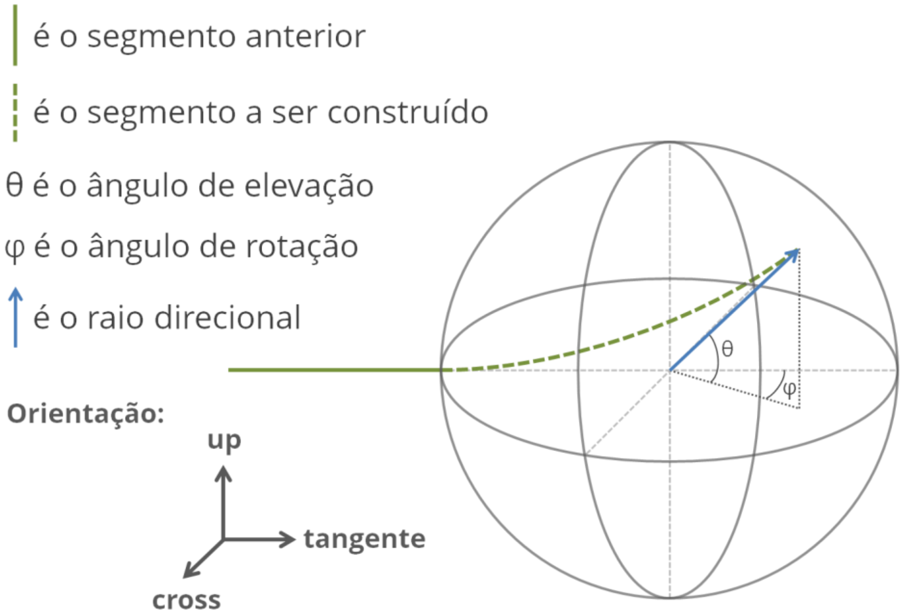
A posição é calculada da seguinte forma, seja $c = 0.551915024494$ e $R_d$ o raio direcional e tangente a tangente inicial:

$$P_{final} = P_{inicial} + c * \|R_d\| * tangente + c * R_d$$

E o ponto de controle da tangente final do seguinte modo:

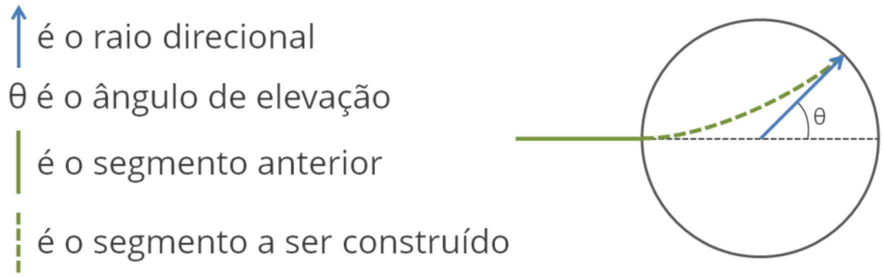
$$P_{controle} = P_{final} - cR_d$$

Observação: $c$ é uma constante para aproximar a Bézier a "um quarto" de círculo.



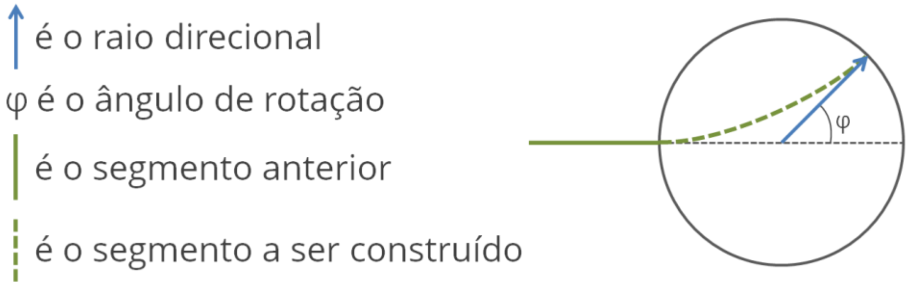

Esquema 2.1.5.1: Esfera imaginária.

A elevação é a primeira rotação feita. O raio direcional é rotacionado θ em relação ao vetor resultante do produto vetorial entre os vetores up e tangente (ou seja, o vetor cross).



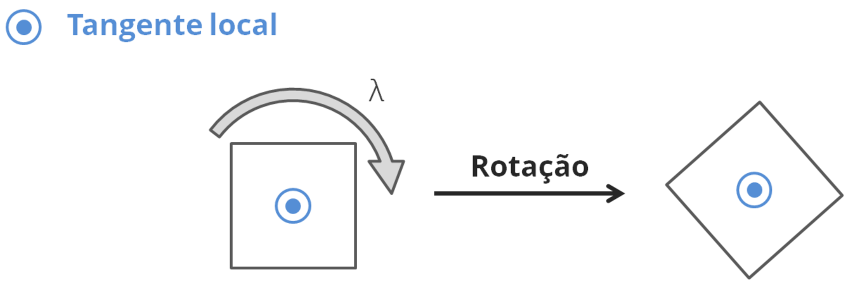

Esquema 2.1.5.2: Elevação do raio direcional.

Depois de elevado, o raio direcional é rotacionado φ ao redor do vetor up.




Esquema 2.1.5.3: Rotação do raio direcional.

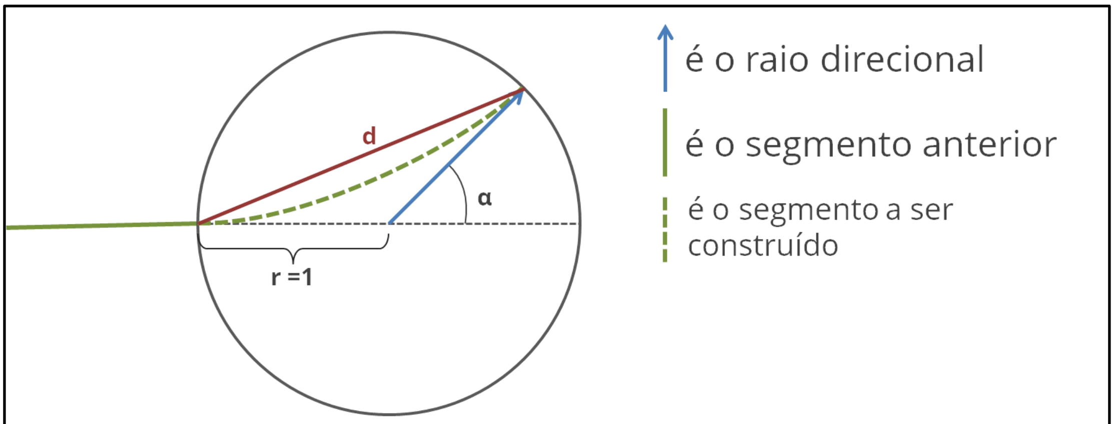
Depois dessas duas rotações, basta apenas "inclinar" o conjunto de pontos ao redor da tangente local a curva que eles pertencem já rotacionada.




Esquema 2.1.5.4: "Inclinação" dos pontos na curva a ser extrusada.

Após essas três rotações, devemos ajustar o comprimento do trilho. Para se manter o comprimento proposto pelo jogador, é necessário fazer o cálculo da norma do raio direcional seguindo as seguintes fórmulas:

Seja "comprimento" o comprimento proposto pelo usuário:




$$\|raio\ direcional\| = \frac{comprimento}{2L} \qquad L = \frac{\alpha d}{2\sin\left(\frac{\alpha}{2}\right)}$$

$$L = \frac{\alpha d}{2\sin\left(\frac{\alpha}{2}\right)}$$

$$d = \|B'(0) + B'(1)\|$$

$$\alpha = \arccos\left(B'(0).B'(1)\right)$$

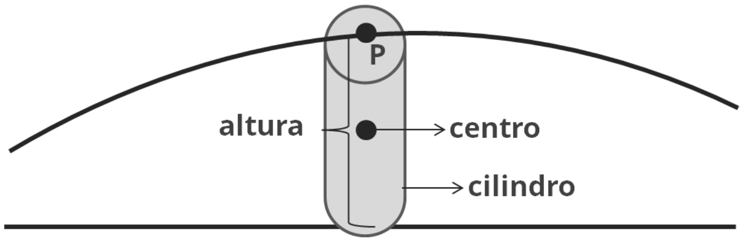
Lembrando que: $\lim_{\alpha \to 0} L = 1$

Esquema 2.1.5.5: Esquema e fórmulas da correção do comprimento.

A dedução dessa fórmula é relativamente simples, basta iniciar com a fórmula do comprimento do arco de uma circunferência. Creio que não seja relevante explicá-la nesse relatório, caso você queira implementar seu construtor, basta apenas aplicar essas fórmulas sem precisar deduzi-las.

### 2.1.6 Suportes

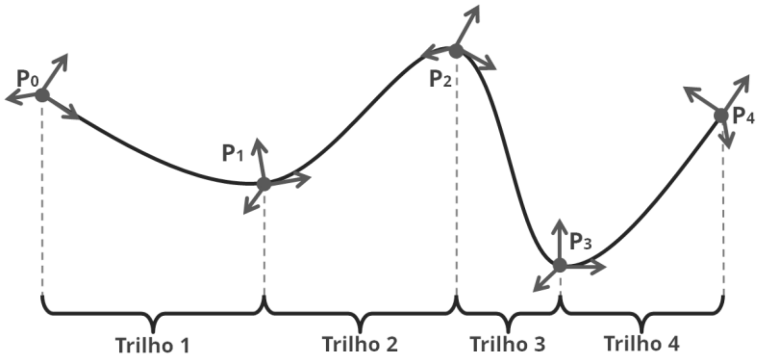
Para adicionar os suportes, devemos selecionar um ponto da curva, calcular sua altura em relação ao chão e instanciar um cilindro com essa altura e centralizado entre o ponto e o chão.



Esquema 2.1.6: Adicionando suporte.

### 2.1.7 Composição

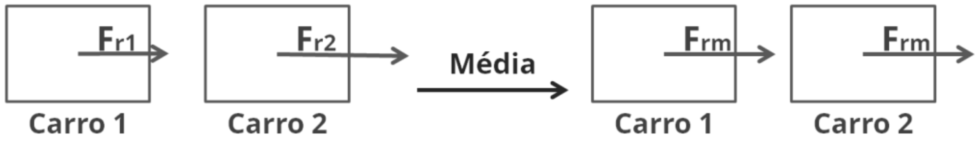
Para finalizar a construção da montanha-russa, devemos compor (ligar) corretamente os trilhos. Para isso, basta colocar a posição final de uma na posição inicial da próxima, atribuindo a orientação final de uma na orientação inicial da próxima.



Esquema 2.1.7: esquema mostrando a composição dos trilhos.

## 2.2 Simulação

Para simular os carros na montanha-russa foi utilizado o método de Euler. Para aplicar as forças nos carros, primeiro se calcula as forças resultantes de cada carro individualmente. Em seguida, se calcula a média das forças resultantes e a aplica a cada carro. Então o conjunto de carros são considerados como apenas um corpo por estarem rigidamente ligados, logo a força aplicada a cada carro é a média das forças resultantes.



Esquema 2.2: aplicação da força resultante média sobre os carros no trilho

A montanha-russa possui 5 tipos de trilhos:

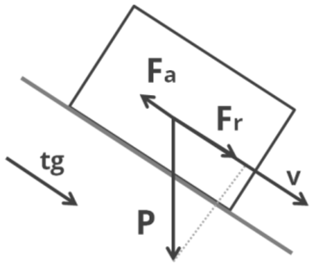
- Plataforma;
- Normal;
- Alavanca;
- Freio;

- Final.

Cada tipo de trilho aplica um tipo de força sobre o carro. A seguir vamos analisar as forças aplicadas por cada tipo de trilho.

### 2.2.1 Trilho  do tipo Normal

Um trilho normal aplica a força da gravidade (**P**) e a força atrito (**Fa**). A força resultante (**Fr**) é definida como:



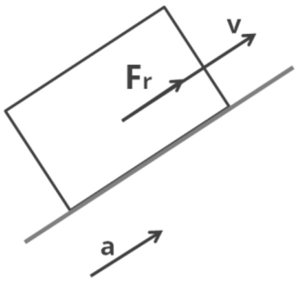

$$\boxed{\begin{array}{l} F_r = (P.tg)tg + F_a \\ P = 9.8 * m \\ F_a = -0.08 * v \end{array}}$$

**tg = tangente do carro**
**v = velocidade do carro**
**m = massa do carro**

Esquema 2.2.1: aplicação de forças sobre o carro no trilho do tipo Normal

### 2.2.2 Trilho do tipo Alavanca

Caso a velocidade escalar esteja maior do que 4 u/s, é aplicada a força do trilho normal. Caso contrário, é aplicada a seguinte força:



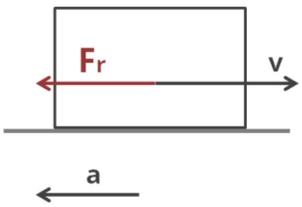

$$\boxed{F_r = a * m}$$

**v = velocidade do carro**
**m = massa do carro**
**a = aceleração = 5 u/s²**

Esquema 2.2.2: aplicação de forças sobre o carro no trilho do tipo Alavanca

### 2.2.3 Trilho do tipo Freio

Caso a velocidade escalar esteja menor do que 5 u/s², é aplicada a força do trilho normal. Caso contrário, é aplicada a seguinte força:




$$\boxed{F_r = a * m}$$

**tg = tangente do carro**
**v = velocidade do carro**
**m = massa do carro**
**a = aceleração = 5 u/s²**

Esquema 2.2.3: aplicação de forças sobre o carro no trilho do tipo Freio

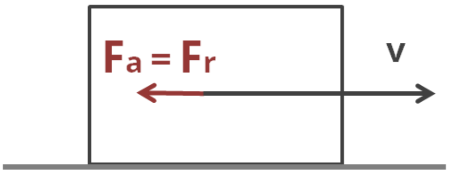
### 2.2.4 Trilho do tipo Plataforma

Caso o carro esteja com velocidade escalar menor do que 4u/s, é aplicado a força da alavanca com aceleração igual a 10 u/s², sendo m a massa do carro. Ou seja, se o carro estiver devagar, ele é empurrado.

Caso contrário, é aplicado a força do freio com aceleração igual a 10 u/s², sendo m a massa do carro. Ou seja, se o carro estiver rápido, ele é freado.

### 2.2.5 Trilho do tipo Final

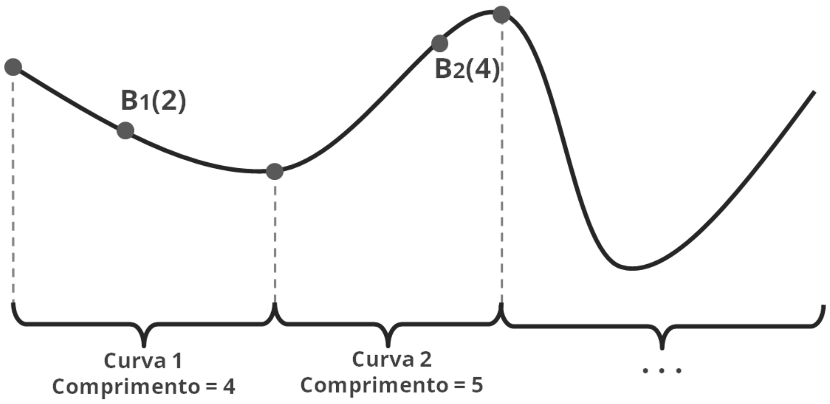
O carro sofre uma desaceleração com magnitude igual a 18 u/s².



Esquema 2.2.5: aplicação de forças sobre o carro no trilho do tipo Final

### 2.2.6 Posição do carro

A posição é em função da composição das curvas de Bézier. É feito algo do tipo Bi(j), onde i representa qual curva estamos posicionando, Bi a função da curva e j a posição local da curva, com j pertencendo a [0, comprimento da curva].



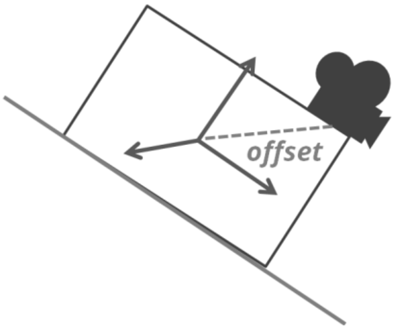
Esquema 2.2.6: Um exemplo de funções Bi(j)

Para calcularmos a posição do carro, basta então alterar o “j” e o “i” de Bi(j). Para isso, como projetamos a força anteriormente, incrementar j com a velocidade escalar do carro. Caso o j atualizado não pertença a [0, comprimento da curva], basta calcular  a curva i’ que ele pertence e atualizar o j para ser relativo a curva i’.

Note que Bi(j) é uma função do R para o R³, Portanto a posição pertence a um espaço unidirecional (1D), ou seja, tecnicamente o carro pode apenas andar “em linha reta”. Logo, é impossível o carrinho sair do trilho, não importa o “delta t” do "*step*". Temos então um sistema de partículas altamente especializado.

### 2.2.7 Primeira Pessoa

Para colocar a câmera em “primeira pessoa”, bastou apenas mudar a orientação e posição da câmera para a do carro e deslocar **localmente** um *offset*.




Esquema 2.2.7: Mudança da orientação e posição da câmera para acompanhar o carro

## 3 Resultados atingidos

Consegui alcançar todos os resultados prometidos na proposta desse projeto e todos os objetivos, completando todo o cronograma. Em suma, foi desenvolvido um construtor de montanhas-russas que simula o carro e ainda oferece a possibilidade de mudar a câmera para o modo "primeira pessoa".

Pode ser visto o projeto pronto neste link:

https://github.com/cesargaspfer/MAC0468_Roller_Coaster

## 4 Conclusões

A implementação de um construtor e simulador de montanhas-russas, embora não seja trivial, é muito interessante pois abrange diversos tópicos utilizados na computação gráfica, como álgebra linear (transformações de base, projeções), cálculo (curvas de Bézier, derivadas), física (simulação dos carros), técnica de extrusão (modelagem da montanha-russa), entre outros tópicos.

Seria muito interessante melhorar o design da montanha-russa, dos carros e do ambiente, salvar a montanha-russa quando o jogador quiser e reestruturar o código para torná-lo mais orientado a objetos. Felizmente a base do código da montanha-russa está pronta, não é mais preciso alterar as equações, transformações e outros tópicos de "baixo nível" (relativos a programar a montanha-russa, não a linguagens de programação). É relativamente fácil adicionar detalhes dado que a base está pronta.

Foi muito proveitoso esse projeto pois finalmente pude aplicar o que apreendi na faculdade até o segundo ano e pude fixar a matéria na minha cabeça de maneira que nunca vou esquecer. Entretanto, o mais importante que aprendi foi lidar com computação gráfica desde a modelagem até a simulação.

Pude ver como ferramentas da matemática são aplicadas em softwares que utilizo todos os dias. Poder ver a teoria por trás da tela do computador (ou do cinema) foi muito inspirador e pretendo ver mais teorias a diante.

## 5 Agradecimentos

Agradeço ao **Marcel Parolin Jackowski** pelas aulas de computação gráfica e pelo código base para a montanha-russa disponível em:

https://github.com/mjck/mac420

Agradeço ao **Matheus Tavares Bernardino** pelos códigos dos exercícios programas feitos na disciplina MAC0420. Alguns trechos deles foram incluídos no projeto atual.

Agradeço a todos que me apoiaram e ajudaram a completar esse projeto.

## 6 Bibliogragia

### 6.1 Citados no relatório

[1] Dicionário do Google consultado em 10/12/2018

### 6.2 Utilizados no programa

https://github.com/mjck/mac420

EPs de 420 de Matheus Tavares Bernardino

Material das aulas de MAC0468 de Marcel Parolin Jackowski

Rotation_matrix function

Approximate a circle with cubic Bézier curves