Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

Semestrální Práce

Programové prostředky řízení

(KKY/PP)

Karel Paukner
A19B0371P

1 Zadání

Zadání semestrální práce z předmětu KKY / PP

Ondřej Severa • 9. 11. 2021 (Upraveno 13. 11. 2021)

Zadání semetrální práce KKY/PP

Na základě přiloženého matematického modelu Míče na válci vytvořte uživatelské rozhraní HMI pomocí jedné z následujících technologií: HTML + SVG + WebSocket; HTML + Canvas + Websocket; Qt + QPainter + REST; Qt + QML + Canvas + REST. Zadání technologie bude náhodně rozděleno.

HMI bude osahovat grafiku, která zobrazí aktuální stav míče na válci, tak aby si nezávislý uživatel dokázal představit v jaké stavu se systém nachází (viz příklad v příloze Mic_na_valci_vzor.jpg). Následně bude doplněno ovládání nastavení počátečního úhlu, "šťouchnutí" do míče, intenzita šumu a reset celého modelu.

Kromě samotné aplikace sepište semestrální práci, která minimálně bude obsahovat:
1. Zadání
2. Stručný popis použitých technologií
3. Popis postupu implementace vaší aplikace (vybrané ukázky zdrojového kódu - klíčové funkce)
4. Základní uživatelský manuál
5. Závěr

Hotovou aplikaci a semestrální práci odevzdáte příslušnému cvičícímu (HTML - Severa, Qt - Faist). Ta bude zkontrolována obsahově a také s ohledem na případné plagiátorství. V případě, že práce bude v pořádku, budete mít nárok na zápočet, který vám zapíše p. Balda u zkoušky.

Popis proměnných:

Fi1_ - Aktuální natočení špulky [rad]
dFi1_ - Aktuální rychlost špulky [rad/s]
Fi2_ - Aktuální poloha středu míče vzhledem ke středu špulky [rad]
dFi2_ - Aktuální rychlost "padání" míče (pohyb středu míče vzhledem ke středu špulky) [rad/s]

Názvy parametrů bloků, které je potřeba měnit z uživatelského rozhraní (pozor názvy jsou Case sensitive)

mic.MP_NUDGE:BSTATE
mic.CNB_DISTRB:YCN
mic.MP_INTEG_RST:BSTATE
mic.CNR_y0:ycn

2 Řešení

2.1 Použité technologie

2.1.1 REXYGEN

REXYGEN je nástroj pro návrh a implementaci komplexních řídicích systémů.

2.1.2 HTML5

HTML je hypertextový značkovací jazyk, který se využívá pro výrobu webových stránek. V dnešní době se používá verze HTML5.

2.1.3 CSS

CSS neboli kaskádové styly jsou používány na nastavení grafické podoby stránek psaných v HTML. V dnešní době se pro zjednodušení práce s CSS používají různé frameworky, nejznámější je například Bootstrap.

2.1.4 JavaScript

Javascript je scriptovací jazyk používaný při výrobě moderních webových stránek. Díky javascriptu je možné udělat interaktivní webové aplikace i příjemnější uživatelské rozhraní doplněné animacemi. V dnešní době se rozrůstá popularita javascriptových frameworků, jimiž jsou například React, Vue a Angular, které usnadňují tvorbu opakovaných částí webů.

2.1.5 SVG

SVG je základním otevřeným formátem pro vektorovou grafiku na webových stránkách. HTML5 umožňuje vložit kód SVG obrázku přímo do kódu HTML webové stránky. Pro naší práci vytvoříme pomocí SVG prvků jednoduchý model našeho systému, který pomocí javasctiptu rozhýbeme.
Kód pro vytvoření našeho modelu:

```
<svg id="svgcanvas" xmlns="http://www.w3.org/2000/svg" version="1.1"
    style="width:700px; height:700px; margin: 0 auto;">
    <path d="M 130 220 C 150 400, 550 400, 570 220" stroke="#d1d1d1" stroke-width="20" fill="none"></path>
    <circle cx="130" cy="220" r="20" stroke="black" stroke-width="1" fill="#d1d1d1" />
    <circle cx="570" cy="220" r="20" stroke="black" stroke-width="1" fill="#d1d1d1" />
    <rect id="sqare" fill="#d1d1d1" stroke="black" x=250 y=370 width="200" height="200"></rect>
    <g id="spool" transform-origin="50% 50%">
        <circle cx="350" cy="360" r="105" stroke="black" stroke-width="4" fill="grey" />
        <circle cx="300" cy="360" r="10" stroke="black" stroke-width="4" fill="grey" />
        <circle cx="400" cy="360" r="10" stroke="black" stroke-width="4" fill="grey" />
        <circle cx="350" cy="410" r="10" stroke="black" stroke-width="4" fill="grey" />
        <circle cx="350" cy="310" r="10" stroke="black" stroke-width="4" fill="grey" />
    </g>
    <g id="ball_circle" transform-origin="50% 50%">
        <circle id="circle" cx="350" cy="360" r="500" fill="none" />
        <g id="ball" transform-origin="50% 50%">
            <circle cx="320" cy="140" r="10" stroke="black" stroke-width="2" fill="grey" />
            <circle cx="380" cy="140" r="10" stroke="black" stroke-width="2" fill="grey" />
            <path d="M 300 180 C 300 230, 400 230, 400 180" stroke="black" stroke-width="4" fill="none"></path>
            <circle cx="350" cy="160" r="95" stroke="black" stroke-width="1" fill="#0074C5" opacity="0.8" />
        </g>

    </g>
</svg>
```

2.2 Implementace

V zadání jsme dostali hotový model v REXYGENU, pro který jsme měli udělat vizualizaci. Pro získání dat z REXYGENU do naší vizualizace jsme použili knihovny rexhmi-vendor.min.js a rexhmi.min.js, které nám přináší důležitou funkci REX.HMI.init, přes kterou můžeme získat všechny proměnné, které potřebujeme.

Tyto javascript knihovny plus i náš javascript main.js, přes který budeme animovat naši vizualizaci, musíme spojit s naší vizualizací v HTML:

```html
<head>
    <meta charset="utf-8" />
    <title>
        KKY/PP Template
    </title>
    <!-- CSS -->
    <link rel="stylesheet" type="text/css" href="css/styles.css">
    <!-- JS -->
    <script type="text/javascript" src="js/rexhmi-vendor.min.js?version=2.50.10-12359"></script>
    <script type="text/javascript" src="js/rexhmi.min.js?version=2.50.10-12359"></script>
    <script type="text/javascript" src="js/main.js"></script>
</head>
```

2.2.1 Inicializace proměnných

Nyní se už můžeme pustit do inicializace proměnných v main.js. Zde pomocí funkce REX.HMI.addItems vybereme proměnné, s kterými budeme dále pracovat.

```javascript
REX.HMI.init = function () {
    // Zde se doplní registrace čtení hodnot s targetu
    REX.HMI.addItems([
        { alias: "Fi1", cstring: "mic.TRND:u1" },
        { alias: "dFi1", cstring: "mic.TRND:u2" },
        { alias: "Fi2", cstring: "mic.TRND:u3" },
        { alias: "dFi2", cstring: "mic.TRND:u4" },
        { alias: "restart", cstring: "mic.MP_INTEG_RST:BSTATE", write: true },
        { alias: "MPoc", cstring: "mic.CNR_y0:ycn", write: true },
        { alias: "stouchnuti", cstring: "mic.MP_NUDGE:BSTATE", write: true },
        { alias: "PORUCHA", cstring: "mic.CNB_DISTRB:YCN", write: true },
        { alias: "poruchalevel", cstring: "mic.SGI:amp", write: true }


    ]);
```

Proměnné, které jen čteme z REXYGENU:

1. Fi1 - Aktuální natočení špulky
2. dFi1 - Aktuální rychlost špulky
3. Fi2 - Aktuální poloha středu míče vzhledem ke středu špulky
4. dFi2 - pohyb středu míče vzhledem ke středu špulky

Proměnné z REXYGENU do kterých budeme zapisovat:

1. restart - slouží pro restart modelu
2. MPoc - slouží pro nastavení počáteční hodnoty natočení míče od špulky
3. stouchnuti - slouží pro šťouchnutí do míče
4. porucha - slouží pro zapnutí/vypnutí šumu
5. poruchalevel - slouží pro změnu intenzity šumu

2.2.2 Duležité funkce

Pro animaci naší vizualizace musíme vždy měnit parametry natočení našich objektů (míč, špulka), vždy když dojde v REXYGENU ke změně těchto proměnných. Na to musíme vytvořit funkci, která se vždy pustí, právě při změně proměnných v REXIGENU:

```
// zobrazeni hodnoty rychlosti otaceni spulky
let dFi1Input = document.getElementById('dFi1');
REX.HMI.get('dFi1').on('change',function(itm){
    let value = radiansPerSecond_to_DegreesPerSecond(itm.getValue());
    // Konverze číselné hodnoty na string s třemi desetinnými místy
    value = value.toFixed(3);
    dFi1Input.innerHTML = value + " deg/s";
});
```

Zde můžeme například vidět funkci, která při změně hodnoty proměnné dFi1 v REXYGENU převede danou hodnotu na stupně a vypíše ji do naší vizualizace.

Dále je pro nás důležitá funkce rotate, která slouží k rotaci elementu podle svého středu v naší vizualizaci:

```
// Rotace elementu dle stredu
function rotate(svgElement, angle) {
    svgElement.style.transform = "rotate(" + angle + "deg" + ")";
}
```

Ke správné funkčnosti této funkce je potřeba u elementu přidat počátek daného elementu (transform-origin = 50% 50%):

```
<g id="spool" transform-origin="50% 50%">
    <circle cx="350" cy="360" r="105" stroke="black" stroke-width="4" fill="grey" />
    <circle cx="300" cy="360" r="10" stroke="black" stroke-width="4" fill="grey" />
    <circle cx="400" cy="360" r="10" stroke="black" stroke-width="4" fill="grey" />
    <circle cx="350" cy="410" r="10" stroke="black" stroke-width="4" fill="grey" />
    <circle cx="350" cy="310" r="10" stroke="black" stroke-width="4" fill="grey" />
</g>
<g id="ball_circle" transform-origin="50% 50%">
    <circle id="circle" cx="350" cy="360" r="500" fill="none" />
    <g id="ball" transform-origin="50% 50%">
        <circle cx="320" cy="140" r="10" stroke="black" stroke-width="2" fill="grey" />
        <circle cx="380" cy="140" r="10" stroke="black" stroke-width="2" fill="grey" />
        <path d="M 300 180 C 300 230, 400 230, 400 180" stroke="black" stroke-width="4" fill="none"></path>
        <circle cx="350" cy="160" r="95" stroke="black" stroke-width="1" fill="#0074C5" opacity="0.8" />
    </g>

</g>
```

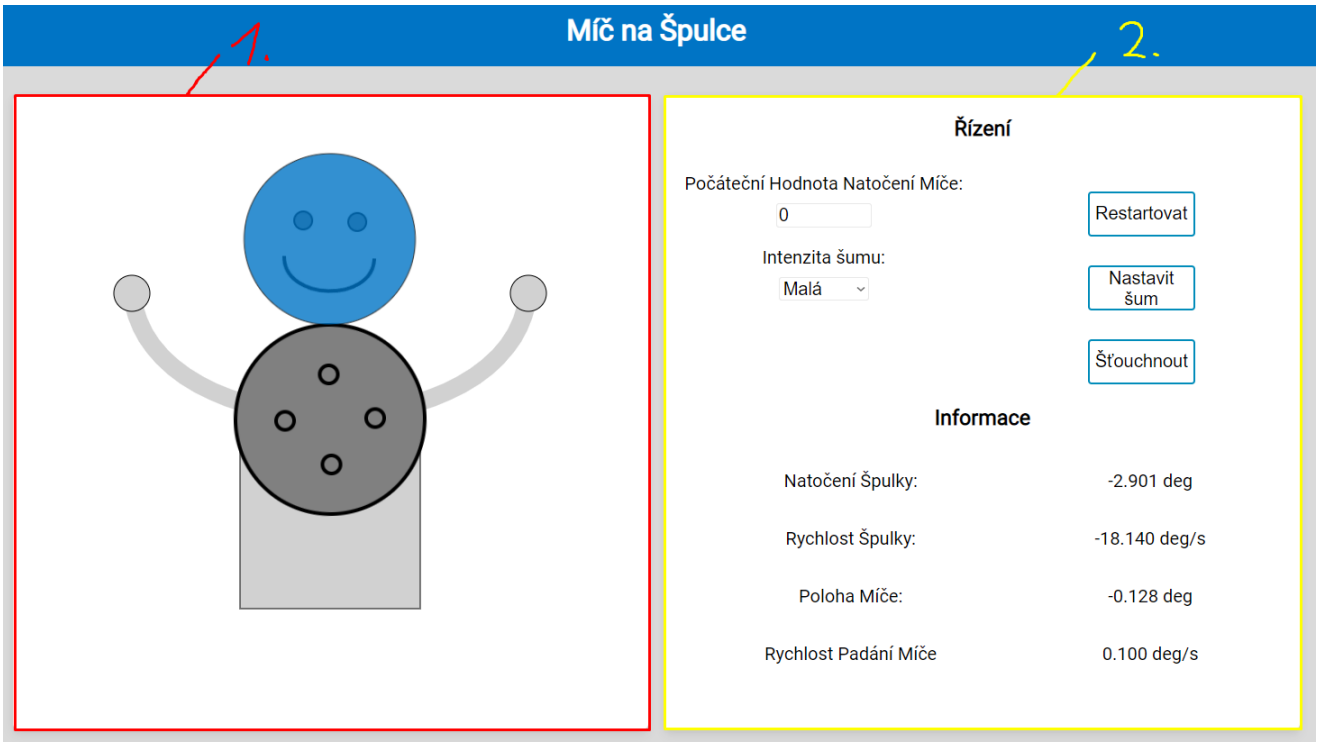
Další použité funkce:

1. funkce radians_to_degrees(radians)
2. funkce degrees_to_radiants(degrees)
3. funkce radiansPerSecond_to_DegreesPerSecond(rad)

2.3 Uživatelský manuál

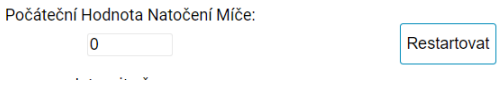
Vizualizace rozdělena na dvě části:

1. Model
2. Část s daty a řízením



Tlačítka:

1. Restartovat - uvede model do počátečního stavu - možné nastavit počáteční hodnotu natočení míče vlevo od tlačítka (ve stupních)



2. Nastavit šum - aplikuje nastavení intenzity šumu- malé/střední/velké/žádné, které lze nastavit vlevo od tlačítka.

3. Šťouchnout - Šťouchne do míče

3 Závěr

K vypracování práce jsem využil Rexygen, HTML5, CSS a JavaScrypt. Jako zdroje informací potřebné k vypracování jsem použil videa z cvičení a stránku https://www.w3schools.com. Pro vytvoření tohoto pdf jsem použil LaTex.