Vážení zákazníci,

dovolujeme si Vás upozornit, že na tuto ukázku knihy se vztahují autorská práva, tzv. copyright.

To znamená, že ukázka má sloužit výhradně pro osobní potřebu potenciálního kupujícího (aby čtenář viděl, jakým způsobem je titul zpracován a mohl se také podle tohoto, jako jednoho z parametrů, rozhodnout, zda titul koupí či ne).

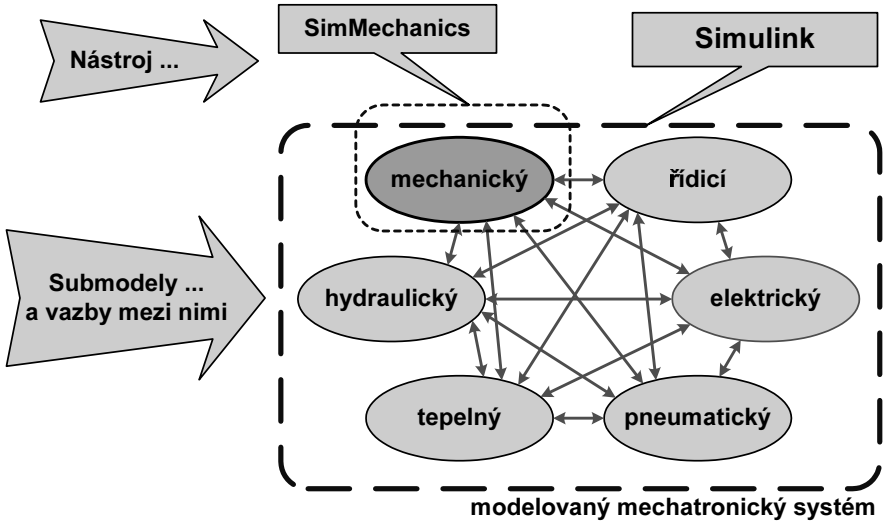
Z toho vyplývá, že není dovoleno tuto ukázku jakýmkoliv způsobem dále šířit, veřejně či neveřejně např. umisťováním na datová média, na jiné internetové stránky (ani prostřednictvím odkazů) apod.

redakce nakladatelství BEN – technická literatura
redakce@ben.cz

Počítačové modelování kinematiky a dynamiky mechanismů je podstatnou součástí moderního přístupu k návrhu technických objektů (strojů, přístrojů, automobilů, letadel). V řadě praktických úloh je možné zanedbat deformace jednotlivých částí mechanismu a pracovat s tzv. **soustavou tuhých těles propojených vazbami**. V anglicky psané literatuře je zaveden termín MBS (multi body system).

Tato kniha je zaměřena na modelování MBS v programu MATLAB/SimMechanics. Díky **integraci do prostředí Simulink** můžeme navíc snadno propojit mechanický model s modelem jiné fyzikální podstaty (elektrický, tepelný, hydraulický, řídicí apod.). Získáváme tak užitečný nástroj pro **mechatroniku**, která je charakterizována jako účelová kombinace mechanických, elektrických a řídicích komponent.

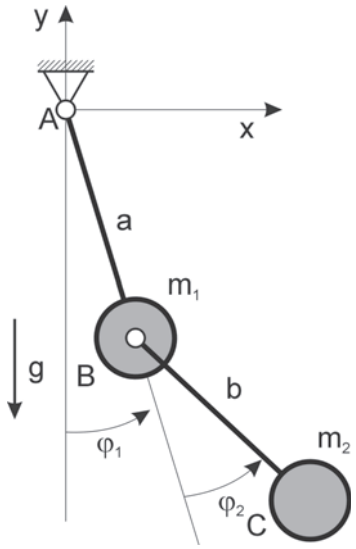
Zásadní výhodou SimMechanics oproti klasickému přístupu k modelování MBS je **automatická tvorba matematického modelu**. Uživatel zadává pouze geometrii a vlastnosti těles a vazeb mezi nimi. Odpadá tak často značně náročný proces „ručního“ odvození pohybových rovnic, úspora času je významná a je možné řešení řádově složitějších problémů.



Obr. 1.1 Model mechatronického systému: submodely a simulační nástroje

1.1 Motivace: dvojité kyvadlo

Efektivitu modelování v prostředí SimMechanics v porovnání s klasickým přístupem naznačíme na příkladu dvojitého kyvadla. Na obr. 1.2 je jeho schéma, víme, že se jedná o soustavu s dvěma stupni volnosti. Pro jednoduchost zanedbáme pasivní odpory a nebudeme uvažovat žádné vnější síly a vliv tíhy zahrneme do potenciální energie.



Obr. 1.2 Schéma dvojitého kyvadla

1.1.1 Odvození pohybové rovnice a model v Simulinku

Klasický přístup spočívá v ručním odvození matematického modelu a následné implementaci v Simulinku. Pro odvození použijeme Lagrangeovy rovnice druhého druhu, přičemž zvolíme relativní zobecněné souřadnice φ_1 a φ_2 podle obr. 1.2. Nejprve vyjádříme kinetickou a potenciální energii:

$$E_k = \frac{1}{2}(m_1 a^2 + m_2(a^2 + b^2 + 2ab\cos\varphi_2)\dot{\varphi}_1^2 + m_2 b^2 \dot{\varphi}_2^2 + 2m_2(b^2 + ab\cos\varphi_2)\dot{\varphi}_1\dot{\varphi}_2)$$
$$E_p = -(m_1 + m_2)ag\cos\varphi_1 - m_2 bg\cos(\varphi_1 + \varphi_2) \tag{1.1}$$

Dále provedeme příslušné derivace podle rovnice

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial E_k}{\partial \dot{q}_i}\right) - \frac{\partial E_k}{\partial q_i} + \frac{\partial E_P}{\partial q_i} = Q_i \ , \quad \text{kde } q_i = \{\varphi_1, \varphi_2\}$$

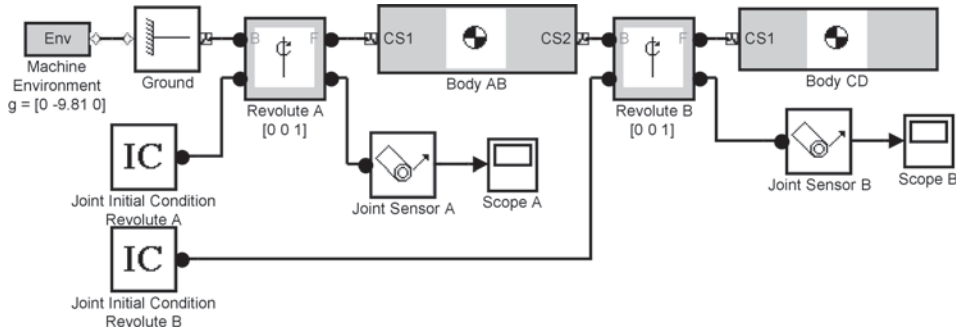
a po úpravách dostaneme dvě poměrně komplikované pohybové rovnice:

$$\begin{aligned}
&((m_1 + m_2)a^2 + m_2 b^2 + m_2 ab\cos\varphi_2)\ddot{\varphi}_1 + \\
&+ (m_2 b^2 + m_2 ab\cos\varphi_2)\ddot{\varphi}_2 - m_2 ab\sin\varphi_2\dot{\varphi}_2^2 - 2m_2 ab\sin\varphi_2\dot{\varphi}_1\dot{\varphi}_2 = 0 \\
&(m_2 b^2 + m_2 ab\cos\varphi_2)\ddot{\varphi}_1 + m_2 b^2\ddot{\varphi}_2 + (m_2 ab\sin\varphi_2)\dot{\varphi}_1^2 + m_2 gb\sin(\varphi_1 + \varphi_2) = 0
\end{aligned} \tag{1.2}$$

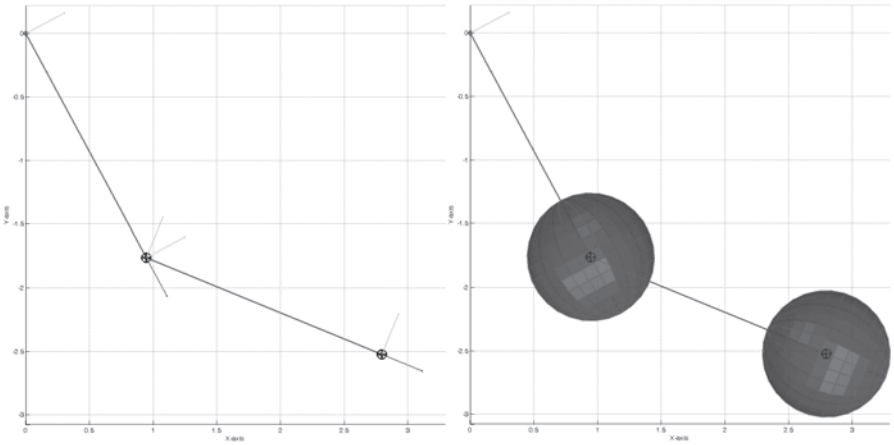
Z těchto výsledných rovnic pak vytvoříme model Simulinku. Vzhledem k tomu, že se v obou rovnicích vyskytuje druhá derivace obou zobecněných souřadnic, nelze je upravit na explicitní tvar. Rovnice lze zapsat v lineárně implicitním tvaru, v Simulinku je tedy nutné pracovat s maticemi.

1.1.2 Model v prostředí SimMechanics

Model stejného problému vytvořený v SimMechanics je na obr. 1.3. Uživatel musí zadat pouze polohu propojovacích bodů a hmotnost obou těles. Mechanismus je pak tvořen bloky `Ground` (definice základního tělesa), `Revolute` (rotační vazba), `Body` (těleso). Pro definici počátečních podmínek v obou vazbách přidáme `Joint Initial Condition`. Chceme-li vykreslit veličiny (polohy, rychlosti, zrychlení, síly, momenty) v obou vazbách, použijeme `Joint Sensor` a `Scope`.



Obr. 1.3 Simulační model dvojitého kyvadla v SimMechanics



Obr. 1.4 Vizualizace simulace dvojitého kyvadla v SimMechanics (vlevo: zobrazení `Convex Hulls`*, vpravo: zapnuté zobrazení ekvivalentních elipsoidů – degenerovaných na koule)*

1.3.2 FAQ – charakteristika SimMechanics pomocí odpovědí na často kladené otázky

V následujících odstavcích najdete odpovědi na otázky, které jsou často kladeny v souvislosti s počátečními pokusy se SimMechanics.

Jaké problémy vlastně mohu v SimMechanics řešit?

SimMechanics řeší úlohy statiky, kinematiky a dynamiky MBS. Díky přímému propojení s bloky běžného Simulinku můžeme řešit i otázky interakce jiných modelů s mechanismem. Příkladem může být úloha *6.15 DC motor připojený na klikový mechanismus*.

Jaký je hlavní rozdíl mezi modelováním dynamiky mechanismů v běžném Simulinku a v SimMechanics?

Rozdíl je zde zcela zásadní: při modelování v Simulinku se software stará pouze o integraci zadaných pohybových rovnic, které uživatel vytvořil např. Newtonovým přístupem nebo Lagrangeovými rovnicemi. Naproti tomu SimMechanics dynamické rovnice sám sestavuje na základě zadané topologie a parametrů MBS. To umožňuje řešení značně složitých a prostorových úloh, u kterých by „ruční“ sestavení rovnic bylo obtížné nebo nemožné.

Umí SimMechanics řešit detekci kolizí mezi jednotlivými tělesy?

Ne. SimMechanics pracuje pouze s parametry těles potřebnými pro dynamickou analýzu (hmotnost, moment setrvačnosti, poloha těžiště) a s vlastnostmi vazeb. Nepracuje (a není nutno zadávat) s tvarem těles. Otevřenost Simulinku ale potenciálně umožňuje provádět detekci kolizí vlastním řešením. Jednoduchý příklad je v úloze *6.13 Model rotační vazby s omezením*.

Jak mohu vizualizovat chování simulovaného mechanismu?

Pro základní animaci mechanismu je použito vykreslování pomocí grafiky Handle Graphics v MATLABu. Tělesa jsou zobrazena jako „konvexní obálky“, není a ani nemůže být respektován jejich skutečný tvar – není totiž v modelu zadán. Můžeme přepínat mezi různými pohledy na mechanismus, ukládat animaci do video souboru apod.

Druhou možností je využití Virtual Reality Toolboxu společně s importovanou CAD geometrií. Výsledkem je podstatně realističtější prostorová vizualizace. Více v kapitole *5.4 Vizualizace mechanismů ve VRML*.

Při spuštění simulace se mi nezobrazuje vizualizace. Proč?

Zobrazení chování mechanismu se nastavuje jednak v bloku `Machine Environment` (záložka `Visualization`) a jednak v nastavení parametrů v menu `Simulation/Configuration Parameters/SimMechanics`.

Jaké další programy potřebuji, abych mohl používat SimMechanics?

MATLAB a Simulink. Pokud požadujeme 3D vizualizaci, je vhodné používat Virtual Reality Toolbox.

Mohu někde získat modely popisované v této knize?

Ano, všechny modely lze stáhnout na webových stránkách nakladatelství BEN – technická literatura.

Jak mohu v SimMechanics namodelovat omezení ve vazbě?

Tento častý požadavek může znít: „rameno robotu se může otáčet pouze v rozmezí 0 až 45°". Jeho řešení ale není snadné. V dynamice MBS a tedy i v SimMechanics nelze nějak jednoduše definovat omezení úhlu v rotační vazbě (v SimMechanics je to `Revolute`). Problém lze vyřešit pouze dosti komplikovaným zavedením fiktivní pružiny s velmi vysokou tuhostí a vůlí. Vysoká tuhost bude ale mít negativní vliv na rychlost simulace. Příklad řešení je v úloze *6.13 Model rotační vazby s omezením*.

Blok Body Sensor se pro lokální souřadnicový systém chová podivně. Proč?

Vysvětlení najdete v kap. 4.3.3 Vlastnosti bloku Body Sensor.

Jak je možné řešit unilaterální (jednostranné) vazby?

SimMechanics neobsahuje speciální bloky pro řešení unilaterálních vazeb. Je-li taková vazba vyžadována, lze ji modelovat na úrovni Simulinku (např. použití bloku Saturation). Poznámku k tomuto problému najdete v kap. *4.1.1 Klasifikace vazeb mezi tělesy v MBS* a *3.2.4 Některé aspekty dynamických formalismů*. Příklad řešení je v úloze *6.13 Model rotační vazby s omezením*.

1.3.3 Příklady využití modelu v SimMechanics

Některé možnosti použití SimMechanics byly přímo či nepřímo zmíněny v předchozím textu. Uveďme je znovu společně s několika dalšími:

Koncepční návrhy

SimMechanics umožňuje rychlé sestavení kinematického nebo dynamického modelu a následné změny parametrů i topologie. V počáteční fázi návrhu můžeme vytvořit několik variant řešení a po porovnání jejich vlastností vybrat optimální. V této fázi nemusíme sestavovat analytický model dynamiky ani geometrický CAD model.

Verifikace „ručně“ sestaveného analytického modelu

V určité fázi návrhu výrobku může být požadován analytický matematický model dynamiky, se kterým bude možné pracovat v reálném čase. Příkladem je použití v pokročilejších řídicích algoritmech. Problémem „ručního“ sestavování rovnic (viz kapitola *3.2 Modelování dynamiky MBS*) je velká pravděpodobnost chyby, která se u složitějších soustav prakticky blíží jistotě. SimMechanics zde s výhodou poslouží pro ověření správnosti „ručně“ formulovaných rovnic.

Model pro návrh automatického řízení

Model vytvořený v SimMechanics (a obecně tedy nelineární) linearizujeme pomocí standardních nástrojů Simulinku. Dostaneme tak stavový model (matice **A**, **B**, **C**, **D**), se kterým dále pracujeme např. v Control System Toolboxu. Navržený regulátor pak můžeme připojit k modelu v SimMechanics a testovat rozdíly v chování nelineární a linearizované soustavy.

Více v řešených úlohách *6.18 Řízení inverzního kyvadla* a *6.19 Řízení nestabilního dopravního prostředku*.

Generátor dat pro aproximační algoritmy

Pomocí opakovaného spouštění modelu v SimMechanics vytvoříme množinu tréninkových resp. testovacích dat pro aproximátor (např. umělou neuronovou síť). Po natrénování může být aproximátor použit samostatně.

Součást virtuálního prototypu

Moderní přístupy směřují k návrhu komplexních počítačových modelů, se kterými lze provádět řadu simulačních experimentů srovnatelných s testováním reálného prototypu. Motivací je pochopitelně úspora času a nákladů. SimMechanics může tvořit mechanickou část komplexního modelu.

2.3 Základní úlohy kinematiky

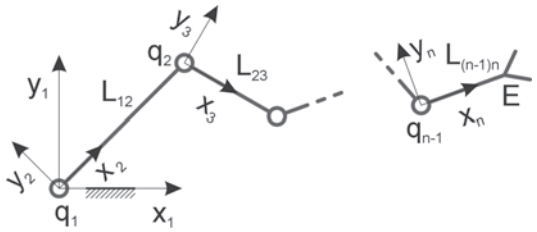
Pro nejjednodušší MBS, kterým je **otevřený kinematický řetězec**, definujeme dvě základní úlohy kinematiky:

- přímá úloha,
- nepřímá úloha.

V případě složitější topologie MBS je situace podobná (otevřený strom, uzavřená topologie, více v kap. *4.1.4 Topologie modelu*). Rozdíl je v použitém algoritmu řešení, což má vazbu na použitou analýzu v SimMechanics (kap. *4.4 Typy analýz v SimMechanics*).

2.3.1 Přímá úloha kinematiky

Uvažujme otevřený kinematický řetězec podle obr. 2.5, který se skládá z (*n-1*) těles, (*n-1*) vazeb a je popsán *n* souřadnicovými systémy.



Obr. 2.5 Mechanismus typu otevřený kinematický řetězec

Pro jednoduchost předpokládejme, že v každém uzlovém bodě je jedna rotační nebo translační vazba, konkrétní natočení nebo posunutí je dáno relativní souřadnicí q_i. Poloha uzlových bodů (vazeb) je dána souřadnicovými systémy podle obrázku (např. $x_2\ y_2\ z_2$). Poloha a natočení (transformace) posledního *n*-tého tělesa vůči pevnému rámu *1* je tedy jednoznačně dána souřadnicemi $q_1, q_2,..., q_{n-1}$.

Chceme-li transformaci koncového tělesa (*n-1*) vzhledem k pevnému rámu *1* vyjádřit matematicky, můžeme s použitím úvah vedoucích k rov. (2.27) snadno psát:

$$\mathbf{T_{n1}} = \mathbf{T_{21}}\mathbf{T_{32}} \dots \mathbf{T_{n(n-1)}} \tag{2.32}$$

Pro polohu libovolného bodu *M* na tělese (*n-1*) v souřadnicovém systému *1* platí:

$$\begin{aligned} \mathbf{r_1^M} &= \mathbf{T_{21}}\mathbf{T_{32}} \dots \mathbf{T_{n(n-1)}}\mathbf{r_n^M} \\ \mathbf{r_1^M} &= \mathbf{T_{n1}}\mathbf{r_n^M} \end{aligned} \tag{2.33}$$

Matici $\mathbf{T_{n1}}$ nazýváme **transformační matice manipulátoru**.

Nyní již můžeme definovat **přímou** (dopřednou, forward) **úlohu kinematiky** takto:

- máme zadány hodnoty zobecněných souřadnic q_i ve všech vazbách,
- hledáme polohu a orientaci souř. systému $x_n\ y_n\ z_n$ (bodu E, koncového efektoru manipulátoru).

Vektor souřadnic ve vazbách bývá zvykem označovat **Q**, vektor polohy a orientace **X**. Formálně tedy můžeme přímou úlohu zapsat

$$\mathbf{X} = f(\mathbf{Q}) \tag{2.34}$$

Uvažujme, že orientaci koncového efektoru E budeme definovat pomocí Eulerových úhlů RPY. Bude tedy platit, že:

$$\mathbf{X} = [x_1^E, y_1^E, z_1^E, \varphi, \vartheta, \psi]^T \tag{2.35}$$

Při pohledu na matici (2.17a) a se znalostí obecného tvaru transformační matice a polohy bodu E můžeme zapsat úplnou podobu přímého kinematického modelu:

$$\begin{aligned}
x_1^M &= \mathbf{T_{n1}}(1,4) \\
y_1^M &= \mathbf{T_{n1}}(2,4) \\
z_1^M &= \mathbf{T_{n1}}(3,4) \\
\varphi &= -\arctan\left(\frac{\mathbf{T_{n1}}(2,3)}{\mathbf{T_{n1}}(3,3)}\right) \\
\vartheta &= \arcsin(\mathbf{T_{n1}}(1,3)) \\
\psi &= -\arctan\left(\frac{\mathbf{T_{n1}}(1,2)}{\mathbf{T_{n1}}(1,1)}\right)
\end{aligned} \tag{2.36}$$

Je vidět, že pro otevřený kinematický řetězec je **řešení** přímé kinematické úlohy **vždy možné v uzavřeném tvaru** (analyticky). Dodejme ještě, že jsme řešili úlohu pro polohu a orientaci koncového efektoru. Řešení úlohy rychlosti a zrychlení je náročnější, pracujeme s maticemi $\mathbf{V_{n1}}$ rov. (2.29) a $\mathbf{A_{n1}}$ rov. (2.31).

2.3.2 Nepřímá úloha kinematiky

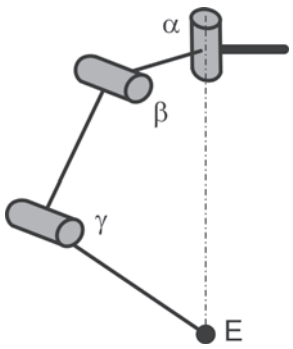
Nepřímá (inverzní, zpětná) úloha kinematiky je, jak název napovídá, úlohou opačnou k přímé:

- máme zadánu požadovanou polohu a orientaci (transformaci) koncového efektoru E,
- hledáme potřebné souřadnice q_i ve vazbách:

$$\mathbf{Q} = f(\mathbf{X}) \tag{2.37}$$

Zamyslíme-li se nad povahou právě definované úlohy, zjistíme, že je podstatně složitější než úloha přímá. Důvodem je:

- **Omezený pracovní prostor** – každý manipulátor má pouze jistý pracovní prostor, často obtížně popsatelného tvaru. Může se lehce stát, že zadáme požadovanou polohu a orientaci, kterou při daných parametrech (rozměrech) manipulátoru nelze realizovat („manipulátor tam nedosáhne“).
- **Dvojznačnost řešení** – i velmi jednoduché manipulátory mohou zaujmout jednu definovanou polohu koncového efektoru dvěma či více způsoby (přitom se nejedná o redundantní manipulátor).
- **Redundantní manipulátor** – pokud je délka vektoru **Q** větší než počet požadovaných souřadnic **X** (což je v prostoru max. 6), mluvíme o redundantním manipulátoru. Pro definovanou polohu a orientaci koncového efektoru pak nelze jednoznačně určit vektor **Q**, možných řešení je nekonečně mnoho.
- **Singulární poloha manipulátoru** – některé (i neredundantní) manipulátory mají ve svém pracovním prostoru oblasti, ve kterých nelze jednoznačně určit některý z prvků vektoru **Q**. Na obr. 2.6 je jednoduchý příklad.



Obr. 2.6 Singulární poloha manipulátoru

Tyto přirozené vlastnosti inverzní úlohy se potenciálně projeví jako značné matematické, programátorské nebo simulační obtíže. Jejich řešení není univerzální a záleží na konkrétní aplikaci. Pro nás, jako uživatele programu SimMechanics, je podstatné o existenci těchto problémů vědět. Lépe pak porozumíme některým chybovým hlášením a zdánlivému zvláštnímu chování simulace. Příkladem může být otázka vhodné polohy mechanismu při řešení inverzní úlohy kinematiky popsané v kap. *6.6 Inverzní úloha rovinného manipulátoru RRR*.

Pro úplnost ještě uvedeme možnosti řešení inverzní úlohy kinematiky:

Analytické řešení

Nejlepší možný případ, kdy vztah (2.37) můžeme vyjádřit v uzavřeném tvaru. Většinou je ale možné pouze u jednodušších úloh.

Numerické řešení

Je založeno na znalosti dopředného modelu rov. (2.34). Iterační metoda uvedená v rov. (2.38) využívá inverze Jakobiánu soustavy podle rov. (2.39).

$$\begin{aligned} \Delta\mathbf{X}_n &= \mathbf{X}^* - \mathbf{X}_n \\ \Delta\mathbf{Q}_n &= \mathbf{J}^{-1}\Delta\mathbf{X}_n \\ \mathbf{Q}_{n+1} &= \mathbf{Q}_n + \Delta\mathbf{Q}_n \\ \mathbf{X}_{n+1} &= \text{DopřednýKinModel}(\mathbf{Q}_{n+1}) \end{aligned} \tag{2.38}$$

kde $\mathbf{X}^*$ je požadovaná hodnota souřadnic a $\mathbf{X_n}$ je okamžitá hodnota souřadnic v n-tém kroku iteračního výpočtu. Jakobián vypočteme takto:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x_1}{\partial q_1} & \frac{\partial x_1}{\partial q_2} & \cdots & \frac{\partial x_1}{\partial q_m} \\ \frac{\partial x_2}{\partial q_1} & \cdots & \cdots & \frac{\partial x_2}{\partial q_m} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial x_n}{\partial q_1} & \cdots & \cdots & \frac{\partial x_n}{\partial q_m} \end{bmatrix} \tag{2.39}$$

Kvůli lepšímu chování v okolí singularit se používá také pseudoinverze Jakobiánu, případně jiné metody. Zajímavé je, že dosti dobře funguje i pouhá transpozice matice **J**. Pro další informace lze doporučit výborný článek [1].

Aproximační řešení

Toto řešení je založeno na předpočítání dostatečného počtu hodnot řešení přímé úlohy a uložení do „tabulky“ (regresní metody, umělá neuronová síť apod.).

V této kapitole si na konkrétních úlohách kinematiky, dynamiky a řízení MBS ukážeme chování většiny simulačních bloků SimMechanics.

Modely je vhodné vždy vytvářet parametricky, tedy:

- parametry definujeme zadáním v m-file (např. `m=10; % hmotnost`)
- v blocích Simulinku nebo SimMechanics pak používáme proměnnou `m` na místo konkrétní hodnoty.

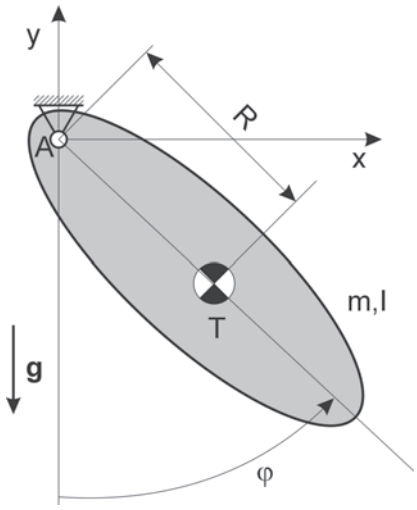
Nejprve pak spustíme definiční m-file, proměnné se zavedou do `Workspace` a následně můžeme pracovat se simulačním modelem. Máme tak k dispozici výpočetní model, jehož parametry můžeme velmi pohodlně měnit z kódu MATLABu. To je jednak přehledné (může se stát, že se jedna daná proměnná vyskytuje v modelu vícekrát a při změnách bychom mohli na některou instanci zapomenout) a také to umožňuje opakované automatické spouštění modelu s různými parametry např. za účelem optimalizace nebo citlivostní analýzy.

Poznamenejme ještě, že spuštění m-file s parametry lze zařídit automaticky definicí v `Model Properties-Callbacks-PreLoadFcn`.

Všechny modely je možné stáhnout na stránkách nakladatelství BEN – technická literatura.

6.1 Fyzikální kyvadlo

Pro první řešenou úlohu jsme zvolili fyzikální kyvadlo. Jde o mechanickou soustavu s jedním stupněm volnosti, která má pro počáteční experimenty vhodné vlastnosti.



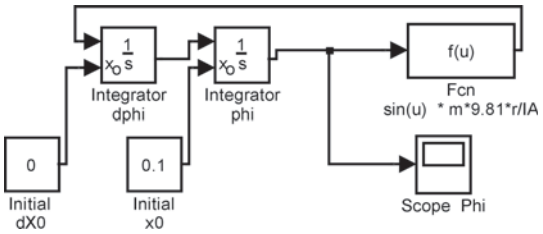
Obr. 6.1 Schéma fyzikálního kyvadla

6.1.1 Odvození pohybové rovnice a model v Simulinku

Předpokládáme pohyb v tíhovém poli Země. Zanedbáme-li tření v závěsu a odpor prostředí, jedná se o konzervativní soustavu. Pohybovou rovnici odvodíme snadno pomocí Lagrangeových rovnic II. druhu nebo uvolněním:

$$\ddot{\varphi} + \frac{mgR}{I_A}\sin\varphi = 0, \qquad I_A = I_T + mR^2 \tag{6.1}$$

Takto jednoduchý systém se snadno modeluje i v prostém Simulinku pomocí dvou integrátorů a bloku `Gain`.



Obr. 6.2 Simulační model kyvadla v Simulinku

6.1.2 Model v prostředí SimMechanics

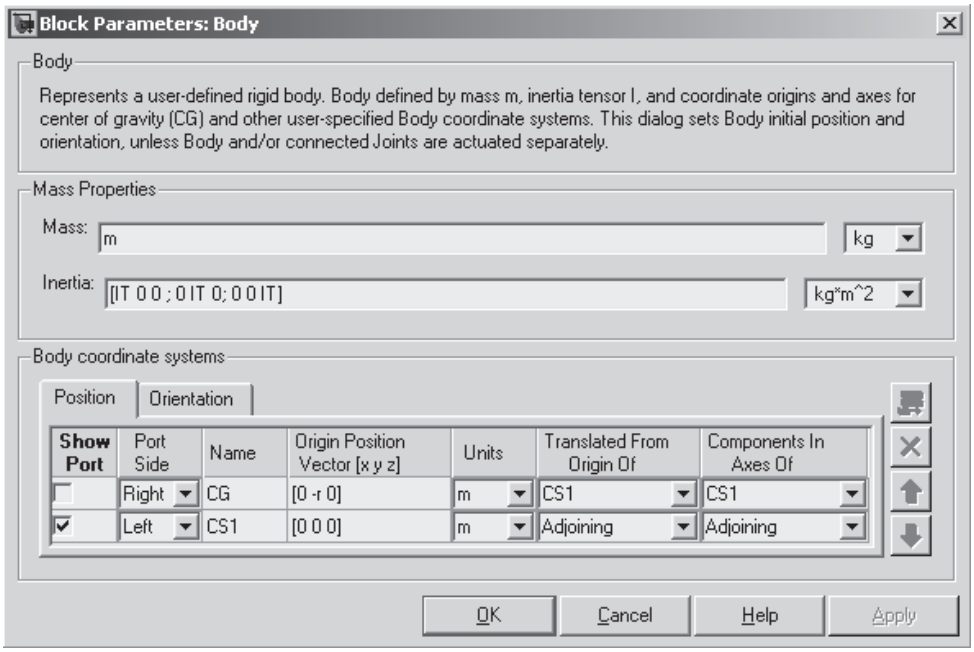
V SimMechanics nebudeme modelovat přímo ODE rov. (6.1), pro vytvoření modelu musíme zadat vlastnosti tělesa a vazby vzhledem k základnímu (pevnému) tělesu.

Model vytvoříme použitím následujících bloků:

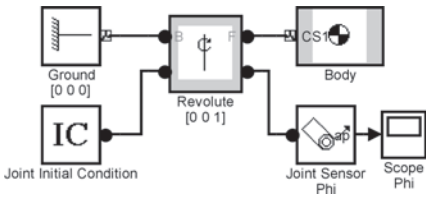
- `Ground` – základní těleso, zadáváme jeho polohu v globálním souřadnicovém systému (`World`).
- `Body` – vlastní kyvadlo, zadáváme polohu těžiště `CG` a polohu alespoň jednoho dalšího bodu, v našem případě bodu A vazby. Nastavení vlastností bodů je uvedeno na obr. 6.3. Záměrně jsme kyvadlo umístili do poněkud neobvyklé vodorovné počáteční polohy.
- `Revolute` – rotační vazba, zadáváme vektor osy rotace, v našem případě [0 0 1].

Výsledný model je na obr. 6.4. Před jeho spuštěním ještě nastavíme zobrazení vizualizace, která je defaultně vypnutá. To provedeme v menu `Simulation/Configuration parameters/SimMechanics` zatržením volby `Display machines after updating diagram` a `Show animation during simulation`.

Tím je simulační model hotov. Je vidět, že i u takto jednoduchého příkladu je návrh v SimMechanics řádově rychlejší než formulace rovnic a návrh v Simulinku.



Obr. 6.3 Okno nastavení vlastností bloku `Body` *(parametry jsou zadány proměnnými, které byly předem definovány v MATLABu a jsou ve* `Workspace`*)*



Obr. 6.4 Simulační model kyvadla v SimMechanics

Další experimentování s modelem již ponecháme na čtenáři samotném. Při tvorbě modelu jsme nezadávali směr a velikost tíhového zrychlení, z jeho chování je ale patrné, že má směr záporné osy *y*. Pokud bychom požadovali jeho změnu, použijeme nastavení v bloku `Machine Environment`. Připojením bloku `Joint Sensor` je možné snímat hodnotu natočení, úhlové rychlosti, vazbových sil a dalších veličin ve vazbě. Blokem `Joint Initial Condition` můžeme měnit počáteční polohu a rychlost kyvadla. Tyto a další bloky a postupy použijeme v následujících úlohách.

6.19.2 Linearizace a návrh řízení

Vstupy a výstupy modelu při linearizaci příkazem `linmod` jsou dány bloky `In` a `Out` ve schématu podle obr. 6.52. Linearizací získáme matice stavového popisu systému:

```
[A,B,C,D] = linmod('model_segway')
```

Předtím ale musíme ze schématu na obr. 6.52 odstranit blok `Continuous Angle`, který obsahuje integrátor a přidává tak další stav do systému. Pak získáme matici **A** s požadovaným rozměrem 4×4. Otázkou zůstává, jaké je pořadí jednotlivých proměnných ve stavovém vektoru, které závisí na vnitřní interpretaci v SimMechanics. Použijeme následující příkazy:

```
[t,x,y] = sim('model_segway')

StateVectorMgr = mech_get_states(x(end,:),
'model_segway/Plant/Machine Environment [0 0 -9.81]')

StateVectorMgr.StateNames
```

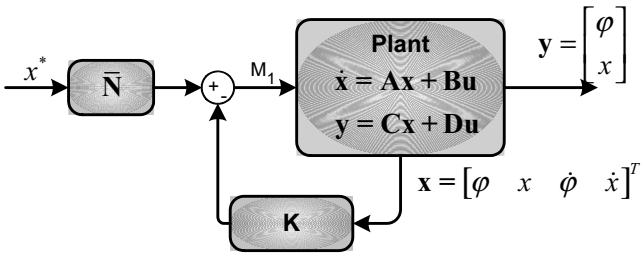
a na příkazové řádce dostaneme

```
ans =

  'model_segway/Plant/Revolute:R1:Position'

  'model_segway/Plant/Prismatic:P1:Position'

  'model_segway/Plant/Revolute:R1:Velocity'

  'model_segway/Plant/Prismatic:P1:Velocity'
```

Vidíme tedy pořadí proměnných ve vektoru **x** a kompletní stavový model můžeme zapsat takto:

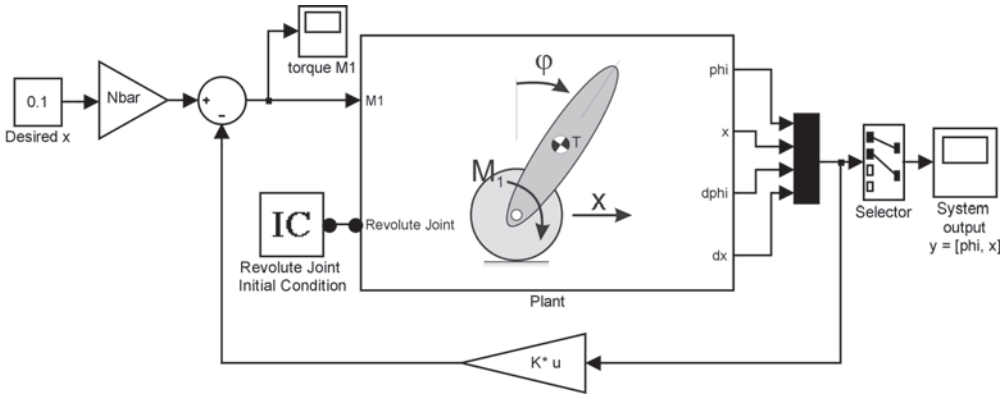
$$\begin{aligned}&\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}\\&\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}\\&\mathbf{x} = \left[\varphi, x, \dot{\varphi}, \dot{x}\right]^T, \quad \mathbf{y} = \left[\varphi, x\right]^T, \quad \mathbf{u} = M_1\end{aligned} \tag{6.38}$$

Pomocí standardních nástrojů pak můžeme navrhnout stavové řízení podle schématu na obr. 6.53. Matici řízení **K** určíme metodou umisťování pólů nebo pomocí LQR [2].

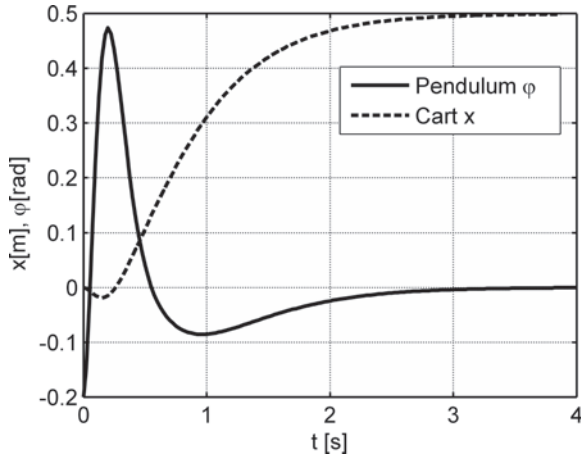


Obr. 6.53 Schéma stavového řízení nestabilního dopravního prostředku

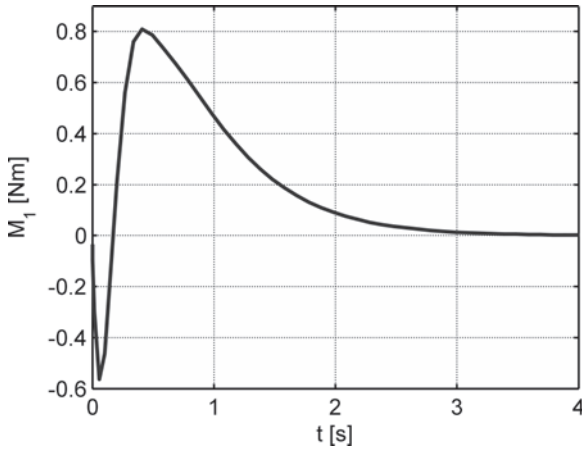
Výsledné schéma simulačního modelu je uvedeno na obr. 6.54. Vstupem do regulované soustavy je požadovaná hodnota pojezdu kola x, přičemž výchylka kyvadla φ je řízena na nulu. Ze stavového vektoru x jsou vybrány výstupní proměnné pro vykreslení (obr. 6.55). Dále můžeme sledovat akční moment M_1. Uvedené schéma by bylo možno dále vylepšovat, např. zahrnout reálné omezení momentu nebo připojit statický či dynamický model elektromotoru.



Obr. 6.54 Simulační model stavového řízení nestabilního dopravního prostředku (subsystém `Plant` *je na obr. 6.52)*



Obr. 6.55 Řízení nestabilního dopravního prostředku – odezva soustavy na požadovanou hodnotu x *= 0,5 m při počáteční výchylce kyvadla* $\varphi = -0{,}2$ *rad. Řízení navrženo pomocí metody LQR.*



Obr. 6.56 Řízení nestabilního dopravního prostředku – velikost akčního momentu při stejných podmínkách jako na obr. 6.55