Teaching Robotics and Mechanisms with MATLAB

Daniela Marghitu
Computer Science and Software Engineering Dept.
Auburn University
Auburn, Alabama 36830
Email: marghda@auburn.edu

Dan Marghitu
Mechanical Engineering Dept.
Auburn University
Auburn, Alabama 36830
Email: marghdb@auburn.edu

***Abstract*—Engineering mechanics involves the development of mathematical models of the physical world. Mechanisms and robots address the motion and the dynamics of kinematic links. MATLAB is a modern tool that has transformed mathematical methods because MATLAB not only provides numerical calculations but also facilitates analytical or symbolic calculations using the computer. The intent is to show using an R(rotation)-R(rotation)T(translation)R(rotation) chain the convenience of MATLAB for theory and applications in mechanisms. The distinction of the study from other projects is the use of MATLAB for symbolic and numerical applications. This project is intended primarily for use in dynamics of multi-body systems courses. The MATLAB graphical user interface enables students to achieve mechanisms programming helping this way the retention in engineering courses. The project can be used for classroom instruction, for self-study, and in a distance learning environment. It would be appropriate for use as an undergraduate level.**

Keywords–MATLAB; symbolic calculations; kinematic chain.

I. Introduction

In this paper, MATLAB is considered for a mathematical equations of a three moving link kinematic chain. The achievements in the robot starting with the development of the recursive Newton-Euler algorithm are given in [1]. Algorithms and equations can be developed using Kane's equations [2] and are of great value. The system kinematics are computed from experimental measurements in [3]. The method for the kinematics of rigid bodies connected by three degrees of freedom rotational joints uses the position measurements. The kinematics of a simulated three-link model and of an experimentally measured motion of human body during flight phase of a jump are discussed. The use of Mathematica and MATLAB for the analysis of mechanical systems is developed in [4] [5] [6]. Educational robotics research studies [7] and our experience of teaching robotics has shown robotics to be one of the best context for teaching computational thinking and problem solving. Habib presented the methodology of integrating MATLAB/Simulink into mechanical engineering curricula [8]. The benefits of integrating MATLAB are the basic concepts, the graphical visualization, and the mathematical libraries. MATLAB software packages for biomedical engineers including nonlinear dynamics and entropy-based methods were presented in [9] and are suitable for an upper-level undergraduate course.

This paper describes the systematic computation of motion for a three link chain using MATLAB. The software combines symbolical and numerical computations and can be applied to find and solve the motion for humans, animals, and robotic systems.

II. Position Analysis

The planar R-RTR chain is considered is shown in Figure 1. The driver link is the rigid link 1 (the link AB). The following numerical data are given: $AB = 0.10$ m, and $CD = 0.18$ m. The angle of the driver link 1 with the horizontal axis is $\phi = 45°$. The constant angular speed of the driver link 1 is 100 rpm. A Cartesian reference frame xy is selected. The joint A is in

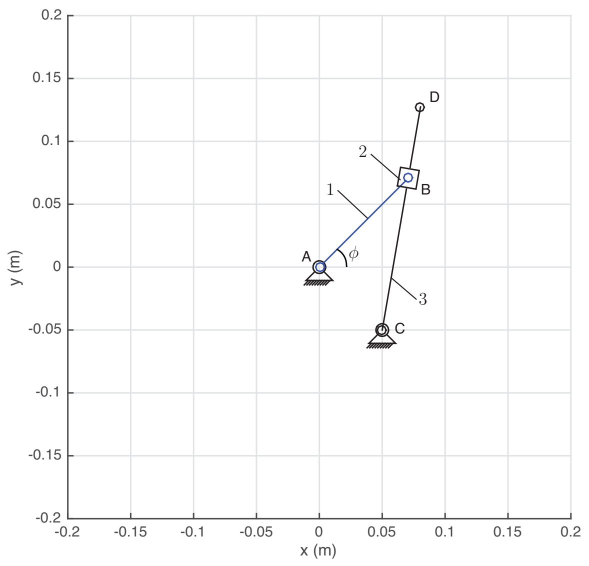

Figure 1. R-RTR chain.

the origin of the reference frame, that is, $A \equiv O$,

$$x_A = 0 \quad \text{and} \quad y_A = 0.$$

The coordinates of the joint C are given

$$x_C = 0.05 \text{ m} \quad \text{and} \quad y_C = -x_C.$$

Position of joint B
The unknowns are the coordinates of the joint B, x_B and y_B. Because the joint A is fixed and the angle ϕ is known, the coordinates of the joint B are computed from the following

expressions

$$x_B = AB\cos\phi,$$
$$y_B = AB\sin\phi. \quad (1)$$

The MATLAB commands for joints A, C, and B are:

```
AB = 0.10 ;  % (m)
CD = 0.18 ;  % (m)
phi = pi/4;  % (rad)
xA = 0; yA = 0;
rA_ = [xA yA 0];
xC =  0.05;  % (m)
yC = -xC;
rC_ = [xC yC 0];
xB = AB*cos(phi);
yB = AB*sin(phi);
rB_ = [xB yB 0];
```

Angle ϕ_2
The angle of link 2 (or link 3) with the horizontal axis is calculated from the slope of the straight line BC:

$$\phi_2 = \phi_3 = \arctan\frac{y_B - y_C}{x_B - x_C}. \quad (2)$$

Position of joint D
The unknowns are the coordinates of the joint D, x_D and y_D

$$x_D = x_C + CD\cos\phi_3,$$
$$y_D = y_C + CD\sin\phi_3. \quad (3)$$

The MATLAB commands for the position of D are:

```
phi2 = atan((yB-yC)/(xB-xC));
phi3 = phi2;
CB = norm([xB-xC, yB-yC]);
ux = (xB - xC)/CB;
uy = (yB - yC)/CB;
xD = xC + CD*ux; yD = yC + CD*uy;
rD_ = [xD yD 0];
```

The components of the unit vector of the vector $\overrightarrow{CD}$ are ux and uy. The numerical results are:

```
% phi = phi1 = 45 (degrees)
% rA_ = [ 0.000, 0.000,0] (m)
% rC_ = [ 0.050,-0.050,0] (m)
% phi2 = phi3 = 80.264 (degrees)
% rD_ = [ 0.080, 0.127,0] (m)
```

The position simulation for a complete rotation of the driver link 1 ($\phi \in [0, 360°]$) is obtained with the MATLAB using a loop command and the graphical representation is shown in Figure 2.

III. Velocity and Acceleration Analysis

The velocity of the point B_1 on the link 1 is

$$\mathbf{v}_{B_1} = \mathbf{v}_A + \boldsymbol{\omega}_1 \times \mathbf{r}_{AB} = \boldsymbol{\omega}_1 \times \mathbf{r}_B, \quad (4)$$

where $\mathbf{v}_A \equiv \mathbf{0}$ is the velocity of the origin $A \equiv O$.
The angular velocity of link 1 is

$$\boldsymbol{\omega}_1 = \omega_1\,\hat{\mathrm{k}} = \frac{\pi n}{30}\,\hat{\mathrm{k}} = \frac{\pi(100)}{30}\,\hat{\mathrm{k}} \;\; \text{rad/s}. \quad (5)$$

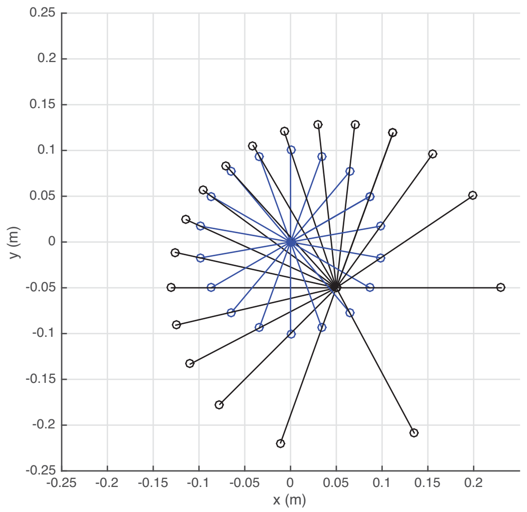

Figure 2. MATLAB representation of the R-RTR chain for a complete rotation of the driver link.

The position vector of point B is

$$\mathbf{r}_{AB} = \mathbf{r}_B - \mathbf{r}_A = x_B\,\hat{\imath} + y_B\,\hat{\jmath} + z_B\,\hat{\mathrm{k}}. \quad (6)$$

The velocity of point B_2 on the link 2 is $\mathbf{v}_{B_2} = \mathbf{v}_{B_1}$ because between the links 1 and 2 there is a rotational joint. The velocity of $B_1 = B_2$ is

$$\mathbf{v}_{B_1} = \mathbf{v}_{B_2} = \begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{\mathrm{k}} \\ 0 & 0 & \omega \\ x_B & y_B & 0 \end{vmatrix}. \quad (7)$$

The acceleration of the point $B_1 = B_2$ is

$$\begin{aligned} \mathbf{a}_B = \mathbf{a}_{B_1} &= \mathbf{a}_{B_2} = \mathbf{a}_A + \boldsymbol{\alpha}_1 \times \mathbf{r}_B + \boldsymbol{\omega}_1 \times (\boldsymbol{\omega}_1 \times \mathbf{r}_B) \\ &= -\boldsymbol{\omega}_1^2 \mathbf{r}_B. \end{aligned} \quad (8)$$

The angular acceleration of link 1 is $\boldsymbol{\alpha}_1 = \dot{\boldsymbol{\omega}}_1 = \mathbf{0}$. The MATLAB commands for the velocity and acceleration of $B_1 = B_2$ are

```
vB1_ = vA_ + cross(omega1_,rB_);
vB2_ = vB1_;
aB1_ = aA_  + cross(alpha1_,rB_) - ...
        dot(omega1_,omega1_)*rB_;
aB2_ = aB1_;
```

The numerical results are:

```
% vB1_ = vB2_ =[-0.740, 0.740,0] (m/s)
% aB1_ = aB2_ =[-7.754,-7.754,0] (m/s^2)
```

The velocity of the point B_3 on the link 3 is calculated in terms of the velocity of the point B_2 on the link 2

$$\mathbf{v}_{B_3} = \mathbf{v}_{B_2} + \mathbf{v}^{rel}_{B_{32}} = \mathbf{v}_{B_2} + \mathbf{v}_{B_{32}}, \quad (9)$$

where $\mathbf{v}^{rel}_{B_{32}} = \mathbf{v}_{B_{32}}$ is the relative acceleration of B_3 with respect to a reference frame attached to link 2. This relative velocity is parallel to the sliding direction BC, $\mathbf{v}_{B_{32}}||BC$, or

$$\mathbf{v}_{B_{32}} = v_{B_{32}}\cos\phi_2\hat{\imath} + v_{B_{32}}\sin\phi_2\hat{\jmath}, \quad (10)$$

where ϕ_2 is known from position analysis. The points B_3 and C are on the link 3 and

$$\mathbf{v}_{B_3} = \mathbf{v}_C + \boldsymbol{\omega}_3 \times \mathbf{r}_{CB} = \boldsymbol{\omega}_3 \times (\mathbf{r}_B - \mathbf{r}_C), \quad (11)$$

where $\mathbf{v}_C \equiv \mathbf{0}$ and the angular velocity of link 3 is $\boldsymbol{\omega}_3 = \omega_3\hat{\mathrm{k}}$. Equations (9), (10), and (11) give

$$\begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{\mathrm{k}} \\ 0 & 0 & \omega_3 \\ x_B - x_C & y_B - y_C & 0 \end{vmatrix} = \mathbf{v}_{B_2} + v_{B_{32}} \cos\phi_2 \hat{\imath} + v_{B_{32}} \sin\phi_2 \hat{\jmath}. \quad (12)$$

Equation (12) represents a vectorial equations with two scalar components on x-axis and y-axis and with two unknowns ω_3 and $v_{B_{32}}$

$$-\omega_3(y_B - y_C) = v_{Bx} + v_{B_{32}} \cos\phi_2,$$
$$\omega_3(x_B - x_C) = v_{By} + v_{B_{32}} \sin\phi_2. \quad (13)$$

The vectorial equation (12) is obtained in MATLAB with:

```
omega3z=sym('omega3z','real');
vB32=sym('vB32','real');
omega3u_ = [ 0 0 omega3z];
vB3B2u_ = vB32*[cos(phi2) sin(phi2) 0];
vC_ = [0 0 0]; % C is fixed
vB3_ = vC_ + cross(omega3u_,rB_-rC_);
eqvB_ = vB3_ - vB2_ - vB3B2u_;
eqvBx = eqvB_(1); eqvBy = eqvB_(2);
```

The solutions of the system are obtained with:

```
solvB = solve(eqvBx,eqvBy);
omega3zs=eval(solvB.omega3z);
vB32s=eval(solvB.vB32);
omega3_ = [0 0 omega3zs];
omega2_ = omega3_;
vB3B2_ = vB32s*[cos(phi2) sin(phi2) 0];
```

The numerical results are:

```
% omega2_ = omega3_ = [0,0, 6.981](rad/s)
% vB32_ = [-0.102,-0.596,-0] (m/s)
```

The acceleration of the point B_3 on the link 3 is calculated in terms of the acceleration of the point B_2 on the link 2

$$\mathbf{a}_{B_3} = \mathbf{a}_{B_2} + \mathbf{a}^{rel}_{B_3B_2} + \mathbf{a}^{cor}_{B_3B_2} = \mathbf{a}_{B_2} + \mathbf{a}_{B_{32}} + \mathbf{a}^{cor}_{B_{32}}, \quad (14)$$

where $\mathbf{a}^{rel}_{B_3B_2} = \mathbf{a}_{B_{32}}$ is the relative acceleration of B_3 with respect to B_2 on link 3. This relative acceleration is parallel to the sliding direction BC, $\mathbf{a}_{B_{32}}||BC$, or

$$\mathbf{a}_{B_{32}} = a_{B_{32}} \cos\phi_2 \hat{\imath} + a_{B_{32}} \sin\phi_2 \hat{\jmath}. \quad (15)$$

The Coriolis acceleration of B_3 realative to B_2 is

$$\mathbf{a}^{cor}_{B_{32}} = 2\ \boldsymbol{\omega}_3 \times \mathbf{v}_{B_{32}} = 2\ \boldsymbol{\omega}_2 \times \mathbf{v}_{B_{32}}$$
$$= 2 \begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{\mathrm{k}} \\ 0 & 0 & \omega_3 \\ v_{B_{32}}\cos\phi_2 & v_{B_{32}}\sin\phi_2 & 0 \end{vmatrix} =$$
$$2(-\omega_3 v_{B_{32}} \sin\phi_2 \hat{\imath} + \omega_3 v_{B_{32}} \cos\phi_2 \hat{\jmath}). \quad (16)$$

The points B_3 and C are on the link 3 and

$$\mathbf{a}_{B_3} = \mathbf{a}_C + \boldsymbol{\alpha}_3 \times \mathbf{r}_{CB} - \omega_3^2 \mathbf{r}_{CB}, \quad (17)$$

where $\mathbf{a}_C \equiv \mathbf{0}$ and the angular acceleration of link 3 is $\boldsymbol{\alpha}_3 = \alpha_3\hat{\mathrm{k}}$. Equations (14), (15), (16), and (17) give

$$\begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{\mathrm{k}} \\ 0 & 0 & \alpha_3 \\ x_B - x_C & y_B - y_C & 0 \end{vmatrix} - \omega_3^2(\mathbf{r}_B - \mathbf{r}_C)$$
$$= \mathbf{a}_{B_2} + a_{B_{32}}(\cos\phi_2 \hat{\imath} + \sin\phi_2 \hat{\jmath}) + 2\ \boldsymbol{\omega}_3 \times \mathbf{v}_{B_{32}}. \quad (18)$$

Equation (18) represents a vectorial equations with two scalar components on x-axis and y-axis and with two unknowns α_3 and $a_{B_{32}}$

$$-\alpha_3(y_B - y_C) - \omega_3^2(x_B - x_C)$$
$$= a_{Bx} + a_{B_{32}} \cos\phi_2 - 2\omega_3 v_{B_{32}} \sin\phi_2,$$
$$\alpha_3(x_B - x_C) - \omega_3^2(y_B - y_C)$$
$$= a_{By} + a_{B_{32}} \sin\phi_2 + 2\omega_3 v_{B_{32}} \cos\phi_2. \quad (19)$$

The MATLAB commands for the calculating α_3 and $a_{B_{32}}$ are:

```
aB3B2cor_ = 2*cross(omega3_,vB3B2_);
alpha3z=sym('alpha3z','real');
aB32=sym('aB32','real');
alpha3u_ = [0 0 alpha3z];
aB3B2u_ = aB32*[cos(phi2) sin(phi2) 0];
aC_ = [0 0 0]; % C is fixed
aB3_=aC_+cross(alpha3u_,rB_-rC_)- ...
dot(omega3_,omega3_)*(rB_-rC_);
eqaB_ = aB3_ - aB2_ - aB3B2u_ - ...
aB3B2cor_;
eqaBx = eqaB_(1); eqaBy = eqaB_(2);
solaB = solve(eqaBx,eqaBy);
alpha3zs=eval(solaB.alpha3z);
aB32s=eval(solaB.aB32);
alpha3_ = [0 0 alpha3zs];
alpha2_ = alpha3_;
aB32_ = aB32s*[cos(phi2) sin(phi2) 0];
```

and the vectorial solutions are:

```
% alpha2_=alpha3_=[0,0,-17.232](rad/s^2)
% aB3B2_=[ 0.505, 2.942,0] (m/s^2)
```

The velocity of D is

$$\mathbf{v}_D = \mathbf{v}_C + \boldsymbol{\omega}_3 \times \mathbf{r}_{CD} = \boldsymbol{\omega}_3 \times (\mathbf{r}_D - \mathbf{r}_C) =$$
$$\begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{\mathrm{k}} \\ 0 & 0 & \omega_3 \\ x_D - x_C & y_D - y_C & 0 \end{vmatrix}. \quad (20)$$

The acceleration of D is

$$\mathbf{a}_D = \mathbf{a}_C + \boldsymbol{\alpha}_3 \times \mathbf{r}_{CD} - \omega_3^2 \mathbf{r}_{CD} =$$
$$\boldsymbol{\alpha}_3 \times (\mathbf{r}_D - \mathbf{r}_C) - \omega_3^2(\mathbf{r}_D - \mathbf{r}_C) =$$
$$\begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{\mathrm{k}} \\ 0 & 0 & \alpha_3 \\ x_D - x_C & y_D - y_C & 0 \end{vmatrix}$$
$$-\omega_3^2 \left[(x_D - x_C)\hat{\imath} + (y_D - y_C)\hat{\jmath}\right]. \quad (21)$$

The velocity and acceleration of D are calculated in MATLAB with:

```
vD_ = vC_ + cross(omega3_,rD_-rC_);
aD_=aC_+cross(alpha3_,rD_-rC_)-...
dot(omega3_,omega3_)*(rD_-rC_);
```

IV. Dynamic Force Analysis

The force of gravity, $\mathbf{G}_1$, the force of inertia, $\mathbf{F}_{in1}$, and the moment of inertia, $\mathbf{M}_{in1}$, of the link 1 are calculated using the MATLAB commands:

```
m1 = rho*AB*h*d;
G1_ = [0,-m1*g,0];
Fin1_ = -m1*aC1_;
IC1 = m1*(AB^2+h^2)/12;
IA  = m1*(AB^2+h^2)/3 ;
alpha1_ = [0 0 0];
Min1_ = -IC1*alpha1_;
```

For the links 2 and 3 there are similar MATLAB statements. The joint forces are calculated using Newton-Euler equations of motion for each link, as depicted in Figure 3. The dynamic

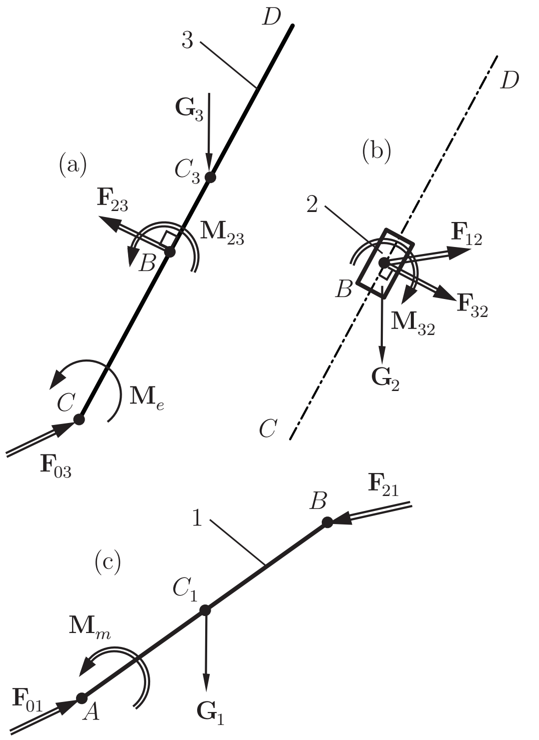

Figure 3. Free body diagrams for each link.

force analysis starts with the last link 3 because the given external moment acts on this link. For link 3, the unknowns are:

```
F23x=sym('F23x','real');
F23y=sym('F23y','real');
F23_=[F23x, F23y, 0];
M23z=sym('M23z','real');
M23_=[0, 0, M23z];
```

Because the joint between 2 and 3 at B is a translational joint the reaction force `F23_` is perpendicular to the sliding direction BC: `eqF23 = (rB_-rC_)*F23_.';` A moment equation for all the forces and moments that act on link 3 with respect to the fixed point C can be written:

```
eqM3C_ = cross(rB_-rC_,F23_)+....
cross(rC3_-rC_,G3_)+M23_+Me_-IC*alpha3_;
eqM3z = eqM3C_(3);
```

There are two scalar equations `%(1)` and `%(2)` with three unknowns `F23x, F23y, M23z`. The force calculation will continue with link 2. For link 2, a new unknown force, the reaction of link 1 on link 2 at B, is introduced:

```
F12x=sym('F12x','real');
F12y=sym('F12y','real');
F12_=[F12x, F12y, 0];
```

The Newton-Euler equations of motion for link 2, are written as:

```
eqF2_ = F12_+G2_+(-F23_)-m2*aC2_;
eqF2x = eqF2_(1);
eqF2y = eqF2_(2);
eqM2_ = -M23_ - IC2*alpha2_;
eqM2z = eqM2_(3);
```

Now there are 5 equations with 5 unknowns and the system can be solve using MATLAB:

```
sol23=solve(eqF23,eqM3z,eqF2x,eqF2y,eqM2z);
F23xs=eval(sol23.F23x);
F23ys=eval(sol23.F23y);
F12xs=eval(sol23.F12x);
F12ys=eval(sol23.F12y);
M23zs=eval(sol23.M23z);
```

The motor moment, `M_`, required for the dynamic equilibrium of the mechanism is determined from:

```
Mm_=IA*alpha1_-(cross(rC1_,G1_)+...
cross(rB_,-F12s_));
```

V. Conclusion

The paper presented a modern tool to teach robotic systems using MATLAB. MATLAB provides numerical and symbolical calculations for an R-RTR kinematic chain. This study is intended for use in mechanism and robotic courses for the undergraduate level.

References

[1] R. Featherstone and D. Orin, "Robot dynamics: Equations and algorithms," in IEEE International Conference Robotics & Automation, San Francisco, CA, 2000, pp. 826–834.

[2] T. Kane and D. Levinson, "Kane's dynamical equations in robotics," International Journal of Robotics Research, vol. 2(3), 1983, pp. 3–21.

[3] J. Lee, H. Flashner, and J. McNitt-Gray, "Estimation of multibody kinematics using position measurements," Journal of Computational and Nonlinear Dynamics, vol. 6(3), 2011, p. 9 pages.

[4] D. Marghitu, Mechanical Engineer's Handbook. Academic Press, San Diego, CA, 2001.

[5] ——, Kinematic Chains and Machine Component Design. Elsevier, Amsterdam, 2005.

[6] D. Marghitu and M. Dupac, Advanced Dynamics: Analytical and Numerical Calculations with MATLAB. Springer, 2009.

[7] J. Davis, B. Wellman, M. Anderson, and R. M., "Providing robotic experiences through object-based programming (preop)," in Proceedings of 2009 Alice Symposium in cooperation with SIGCSE, ACM, Durham, NC, 2009.

[8] M. Habib, "Enhancing mechanical engineering deep learning approach by integrating matlab/simulink," Int. J. Engng. Ed., vol. 21, 2005, pp. 906–914.

[9] J. Semmlow and B. Griffel, Biosignal and Medical Image Processing. CRC Press, 2014.