Airo International Research Journal June,2015
Volume V, ISSN: 2320-39714

PERIODIC, APERIODIC AND SPORADIC TASKS INDENFICATION AND ESTIMATION USING IMAGE PROCESSING IN REAL TIME EMBEDDED AUTOMOTIVE ELECTRONIC SYSTEMS

Shaik Saidulu
Associate Prof, ECE
GNIT-HYDERABAD
Cell: 91-9603066613
Email: sk.saidulu@gmail.com

Dr. Yash Pal Singh
Prof, Dept of ECE, SunRise University
Email: ypsingh10@rediffmail.com

Abstract: - *This paper deals with the automotive electrons system in driver less car. Automotive electronics is popular in now way days. The most of the automotive protocols and systems designed based on real time embedded systems. These are very sophisticated systems and take the dynamic decisions in the real time world based on the parameters passed from the sensor inputs and activate the output devices like actuators with high efficiency. The Scheduling algorithm is one of the most important portions of embedded operating systems especially for real-time embedded operating systems. The performance of scheduling algorithm will influence the performance of the whole system. Real time embedded operating system needs better response time for real-time process; it is more rigid on response time for hard real-time embedded operating system. If any task failure in real time system gives a major damage like life.*

Keywords: - *Real-Time System, Automotive Electronics, Image Processing, Raspberry PI, Task Scheduling, periodic Process, Aperiodic Process, Sporadic process*

1. Introduction: - Today, embedded electronics have stronghold in every space of the society. To full fill the demand of people lots of research and development are going on in this domain. In Particular, frequently changing needs of people gives rise to Reconfigurable real time system. Those systems are typically embedded devices implemented on Microcontroller based platforms which support reconfiguration in the resources even during run-time. This resource reconfiguration is managed by the Real time operating system (RTOS). RTOS is operating systems designed for real time embedded systems. RTOS is a collection of different real time algorithms for managing resources, schedule the tasks and provide lots of other functionality for real time applications. In this paper we propose a novel Multi shape task scheduling algorithm for scheduling real-time task and compare with the conventional algorithms [1]. Real time systems are those systems which supports real-time constraints. Real time systems can be categorized into three categories i.e. hard real time systems, soft real time systems, and firm real time systems. Hard real time systems are those systems in which time delay in response can results potential failure to the system or the loss of human life. In general there is a cost function associated with the system. Many of these systems are considered to be safety critical. Soft real time system is those system where deadline overruns are tolerable, but not desired. There are no hazardous consequences of missing one or more deadlines. Firm real time systems are mix of hard real time and soft real time system. Here infrequent deadline missing is tolerable but it degrades the performance of the systems. To avoid the deadline missing, we need to schedule the task effectively [2]. To solve a scheduling problem, we try to find a schedule for the tasks to execute in such a way so that tasks can be completed before the deadline. Scheduling can be used in any domain where is the limited number of resources to be scheduled efficiently. This efficiency means optimizing desired criteria. Such criteria could be to minimize the schedule length, to maximize the resources utilization ratio to maximize the number of accepted tasks. In this paper, we analyzed algorithms for the periodic tasks. A periodic task is an infinite sequence of jobs with periodic ready times, where the deadline of a job could be equal to, greater than or less than ready time of the succeeding jobs [3]. Scheduling algorithms can be classified in, Uni-processor vs Multiprocessor Scheduling, Soft real time vs hard real time Scheduling, Static vs Dynamic Scheduling, Fixed vs Dynamic Priority Scheduling, Pre-emptive vs Non pre-emptive Scheduling. So for real time systems different Task scheduling algorithms have been studied to avoid the deadline missing. Some of the popular real time task Scheduling algorithms are Earliest Deadline first (EDF), Fixed priority Task scheduling, rate monotonic (RM), Deadline monotonic (DM) etc [4] [5]. Rate monotonic Scheduling algorithm is mostly researched, reviewed, and analyzed algorithm. It is a priority driven algorithm in which priorities are assigned according to the cycle period of the job i.e. the task which has less cycle duration, get the higher priority. So for periodic Tasks, priority of the all instances of the task is known before the arrival and it will be same for all instances. It supports pre-emption and work well on uni-processor system [6]. In DM is a scheduling algorithm, priorities are assigned according to relative deadline Di i.e. higher priority is assigned to the task which has less deadline. It will be fixed for all the future instances. In general, DM is optimal for systems that consist of pre-emptive synchronous independent tasks whose relative deadline is less to their

eid(i≤Pro)D with periodic, aperiodic and sporadic tasks

systems [7]. EDF algorithm is a dynamic priority algorithm; the priorities are assigned based on the value of relative deadline of the tasks in real time. The task with nearest deadline is given highest priority and it is selected for execution. EDF could be applied to various tasks models (preemptive, non preemptive, periodic, non-periodic, etc.).EDF has also been shown to be optimal in the case of non-periodic tasks. EDF scheduling outperforms RM and produces less pre-emption compared to RM and it is very fast.

Criteria for a good scheduling algorithm:

- fairness: all processes get fair share of the CPU
- efficiency: keep CPU busy 100% of time
- response time: minimize response time
- turnaround: minimize the time batch users must wait for output
- throughput: maximize number of jobs per hour

2. APPROACHE: Real-time computer systems are being ubiquitously deployed in many mission and safety critical applications, and are increasingly becoming the backbone of most modern cyber-physical systems, e.g., autonomous vehicles. They are typically based on contemporary uniprocessor and multiprocessor platforms which support performance enhancing hardware, e.g., caches and instruction pipelines to prefetch data and instructions that significantly improve average system performance. Preemptively scheduling hard real-time tasks on such platforms typically imply non-negligible preemption and migration related overheads, potentially causing deadline misses. Consequently, the deployment of such modern processors in real-time systems requires a careful analysis of the resulting hardware-software ecosystem. High preemption and migration related overheads are considered to be an emerging problem in many real-time applications in autonomous vehicles where data intensive operations, such as image processing for vehicular vision systems, form a critical part of the software. On the other hand, as pointed out by Short [5], non-preemptive scheduling is often favored for applications with severe resource constraints due to its low memory requirements and simple implementation. However, non-preemptive scheduling has received less attention as compared to preemptive scheduling. The basis for our scheduling algorithm is the EDF scheduling policy [17]. The EDF is not always optimal for our synthesis problems. The following summary of observations, which can be easily proved or are already proved, guided us to develop an effective and efficient heuristic for the EDF-based task level scheduler.

3. Source code:

```
#!/usr/bin/python
from time import sleep
import serial
import sys
import RPi.GPIO as GPIO
out = serial.Serial('/dev/ttyAMA0',38400)

#use GPIO4 has servo reset
ServoReset = 7


if name == ' main ':
    if len(sys.argv) is 3:
        Servo = int(sys.argv[1])
        Position = int(sys.argv[2])
    else:
        GPIO.setmode(GPIO.BOARD)
        GPIO.setup(ServoReset,GPIO.OUT)
        GPIO.output(ServoReset,False)
        GPIO.output(ServoReset,True)
        sleep(0.1) #write to an unused servo just to sync
        out.write('\x80\x01\x04\x7f\0x11\0x11')
        print('Servo initialized!')
        sys.exit()

out.write('\x80\x01\x04' + chr(Servo) + chr(Position >>7) + chr(Position & 0x7f))
```

```
#!/usr/bin/env python
#this program will control the robot car using keyboard keys
#Ce programme permet de contrôler la voiture robot avec des touches de clavier
# keys function
#
# car move ( mouvement du véhicule)
# w : forwad (avancer)
# a : left (tourner à gauche)
# s : right (tourner à droite)
# z : reverse (reculer)
#
# Webcam
# i : up (caméra vers le haut)
# j : left (caméra vers la gauche)
# k : right (caméra vers la droite)
# m : down (caméra vers le bas)
#
#
# Jaw (la pince)
# o : open (ouvrir)
# p : close (fermer)
#
#
# v : turn all servos off (Fermer tout les servos)
#
#
# Sound ( le son)
# keys 0 to 9 play different sound
# les clef de 0 à 9 jouent différents son.
(0 = sirène ambulance , 9= R2d2,
#
# any other key will stop the movement of the car
# toute clef non déclaré arête le véhicule

from time import sleep
import RPi.GPIO as GPIO
import sys, termios, atexit
from select import select
import serial
import ossaudiodev
import wave
from threading import Thread

out = serial.Serial('/dev/ttyAMA0',38400)
```

Results:

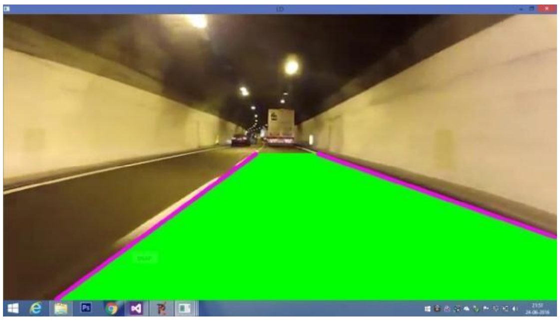

Fig: 1path estimation and object detection

The above simulation result is shown in green color path concern the no objects and safe drive path estimation.

Fig 2: identification of path curvature

The above fig simulation identified the curvature in a path and estimated the direction of curvature.

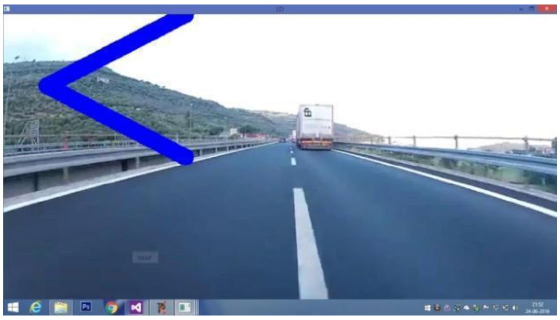

Fig 3: lane crossing identification

The above fig indicates that the line crossing of the vehicle and green color path is available for safe ride. Based on the above three results the path estimation and identification of automotive vehicle has been observed.

CONCLUSION:

Earliest Deadline First algorithms are presented the least complexity according to their performance many of the supposed „problems‟ that have type of scheduling technique. It is clear that earliest deadline first is the efficient scheduling algorithm if the CPU utilization is not more than 100% but does scale well when the system is overloaded. In the experimental environment EDF scheduling algorithm can meet the needs of real-time applications.

REFERENCES:

[1]. Jag beer Singh, Satyendra Prasad Singh "An Algorithm the Time Complexity of Earliest Deadline First Scheduling Algorithm in Real-Time System", International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 2, No.2, pp 31-35 February 2011

[2]. Jashweeni Nandanwar " Real Time Task Sc International Journal of Computer Science Issues (IJCSI), Vol. 9, Issue 6, No 1, pp 335-339 November 2012 ISSN (Online): 1694-0814

[3]. Miss. Rina V. Bhuyar1 Dr. D. G. Harkut "Adaptive Neuro Fuzzy Scheduler for Real T International Journal of Advanced Research in Computer Science and Software Engineering Volume 4, Issue 2, pp 393-396 February 2014 ISSN: 2277 128XI.

[4]. Dr. D. G. Harkut, Anuj M. Agrawal, "Comparison of Diffe Scheduling Algorithms International Journal of Advanced Research in Computer Science and Software Engineering Volume 4, pp 1236-1239, Issue 7, July 2014.

[5]. Michael Short- "The Preemptive, Deadline-driven Scheduling in Real-time Embedded Systems", Proceedings Congress on Engineering 2010 Vol I WCE 2010, pp June 30 - July 2, 2010, beenLondon,U.Kattributed. to this

[6]. Umm-I-aiman &Sherafzal khan "review of different optimal performance of multi-processors", VFASTAugust,

Transactions on 2013 Volume 1, Number 2, pp 7-11,July-August 2013.

[7]. Sanjoy baruah-driven [15]. “PriorTangyin,ty-Time “RealOperating Scheduling of periodic task systems on multiprocessors”,-201,Guide”ppChina188Electric Power Press, 2003.

[8]. Hamza Gharsellaoui,NewOptimal Solutions for Real-Time Reconfigurable Periodic Asynchronous OS Tasks with Minimizations of Response Times”,-18. pp

[9]. R. Kalpana,An S.[17] . KeerthikaRalfSteinmetz,"Analyzing“ the Efficient Non-Preemptive Algorithm for Soft Real-Time Systems using Domain Cluster–Group ,EDF” International Journal of Computer Applications (0975 –8887) Volume 93 –No.20, pp 1-7 May 2014.

[10].Ishan Khera, Ajay Kakkar, “Comparative Study of Scheduling Algorithms for Real Time Environment”, International j urnal of computer applications, Vol 44, No. 2 pp15-22, April 2012.

[11] Leey, Miodrag Potkonjaky and Wayne Wolfz “Synthesis - of Hard Time Application Specific Chunho yComputer Science Dept., University of California, Los Angeles, CA zDept. of Electrical Engineering, Princeton University, Princeton, NJ.

[12]. Miao Liu et.all, “On Improving Real Time Interrupt Latencies of Hybrid Operating Systems with Two-Level Hardware Interrupts”,IEEE Transactions On Computers, Vol. 60, No 7, July 2011, pp. 978-991.

[13]. Mian Dong and Lin Zhong, “Power Modeling and Optimisation for OLED Displays”, IEEE Transactions on Computers,Vol. 11, No 9, September 2012, pp. 1587-1599.

[14]. Q. Li and C-Time. Yao, Concepts for Embedded CMP Books, 2003.

System Application development July 2002.

[16].C. L“. Liu, J. W. Layland, "Scheduling Algorithm for Multiprogramming in a Hard Real-Time Environment," Journal of the 1ACM, 20-91, Jan, 1973.

Multimedia Operating System," IEEE Multimedia, Spring, 1995.

[18]. C.L. Liu, J.W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment", Journal of the ACM, 20(1) pages 46-61, 1973.

Author’s Profile:

***Mr. Shaik Saidulu**, His completed UG&PG Engineering from JNTUH. He is a Ph. D Research Scholar from SUNRISE University, Alwar. He has been published 35 international journals and conference Papers in various hi-indexed Journals so far. His projects awarded prizes i various capitations. His research interests are Soft Computing, IoT, Embedded Networking, Processor Architectures and Intelligent Systems.*

Contact Details:
Shaik Saidulu
Plot N0-101, Road No-3,

Suryodaya Colony, LB Nagar, HYD-500074.
Contact: 91-9603066613
Email: sk.saidulu@gmail.com

***Dr.** Yash Pal Singh is having vast 31 years of experience in Teaching, Research and Industry. He got the Ph.D from poona University, PUNE in the year 1987. He is the member & incharge of many committees formed by AICTE, MHRD, Govt of India etc.* Project In- charge Govt. Of Delhi to generate Internal Revenue for the Govt. he had worked as Programme Coordinator to impart training to the Officers and officials of various Deptt of Delhi of Govt. He is Published 86 National & 210 International papers in Various Hi-Indexed Journals. He registered as Ph.D guide; Subject expert, Guest Lecture and key note speaker of various well known universities. As on 16 Ph.d guided and 11 ongoing.

Consultancy Projects:

1) U.T.S. system, Northern Railway,Delhi.
2) Networking Support for IMPRESS System for IRCTC.
3) Training on soft skills for P.G.T."S, Vice Principal for Directorate of Education, Govt of Delhi.
4) Indian Army for the Microwave tubes and other allied antennas components.

Books Published:

(I) Electronics & Materials.
(II) SEMICON DUCTOR DEVICES
(III) Introduction to Computers. (IV) BASIC ELECTRONICS
(V) Laser and its application
(VI) Wireless & Mobile Communications

Contact Details:

Dr.Yash Pal Singh
R.Z.E.-II/24,New Roshanpura,
Najafgarh, New-Delhi- 43 (INDIA)
Mob:**09810516527** & **011-25015911**
E-mail: ypsingh10@rediffmail.com