Design, Implementation and Analysis of Wireless Model Based Predictive Networked Control System over Cooperative Wireless Network

by

M. ALPHAN ULUSOY

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabanci University
Spring 2009

Design, Implementation and Analysis of Wireless Model Based Predictive Networked Control System over Cooperative Wireless Network

APPROVED BY

Assoc. Prof. Dr. Özgür GÜRBÜZ .....
(Thesis Advisor)

Assist. Prof. Dr. Ahmet ONAT .....
(Thesis Co-advisor)

Assist. Prof. Dr. Volkan PATOĞLU .....

Assoc. Prof. Dr. Albert LEVİ .....

Assist. Prof. Dr. Onur KAYA .....

DATE OF APPROVAL: .....

To my loving family

&

friends

Acknowledgments

I owe my deepest gratitude to my thesis advisors Associate Prof. Dr. Özgür Gürbüz and Assistant Prof. Dr. Ahmet Onat for their exceptional trust, never-ending support, encouragement and guidance. This thesis can be completely attributed to them as —besides everything else— they are the ones who introduced me to the cornerstones of this thesis during my undergraduate studies at Sabanci University and lit the first sparks in my mind that eventually led to this work.

This thesis would not have been possible without the endless support of my family; my mother's refreshing early morning teas, my father's skillful hands that constructed the test platform rack and my brother's sincere support. I do not believe I can ever pay them back.

Many times during the course of this work I have felt lost, exhausted and found myself turning in circles. If it was not for Melda Şener, probably the single most important person in my life, I would not be able to come this far. She has always been my beacon helping me find my way with her ideas, guidance, love, care and support.

I am also indebted to my close friends and remarkable colleagues for their support, encouragement and all the fun times we had which filled this challenging period of my life with moments to remember.

This thesis was supported by scholarships from Sabancı University and the Scientific and Technological Research Council of Turkey (TÜBİTAK).

Design, Implementation and Analysis of Wireless Model Based Predictive Networked Control System over Cooperative Wireless Network

M. Alphan Ulusoy

Electronics Engineering, MS Thesis, 2009

Thesis Advisors: Assoc. Prof. Dr. Özgür GÜRBÜZ,
Assist. Prof. Dr. Ahmet ONAT

Keywords: wireless networked control systems, model based predictive control, cooperative medium access control protocols

Abstract

Owing to their distributed architecture, networked control systems are proven to be feasible in scenarios where a spatially distributed control system is required. Traditionally, such networked control systems operate over real-time wired networks over which sensors, controllers and actuators interact with each other. Recently, in order to achieve the utmost flexibility, scalability, ease of deployment and maintainability, wireless networks such as IEEE 802.11 LANs are being preferred over dedicated wired networks. However, basic networked control systems cannot operate over such general purpose wireless networks since the stability of the system is compromised due to unbounded delays and unpredictable packet losses that are typical in the wireless medium.

Approaching the wireless networked control problem from two perspectives, this thesis proposes a novel wireless networked control system and a realistic cooperative medium access control protocol implementation that work jointly to achieve decent control even under unbounded delay, bursts of packet loss and ambient wireless traffic. The proposed system is implemented and thoroughly evaluated on a dedicated test platform under numerous scenarios and is shown to be operational under bursts of packet loss and ambient wireless traffic levels which are intolerable for basic networked control systems while not being hindered by restraining assumptions of existing methods.

İşbirlikli Kablosuz Ağ Bağlantılı Modele Dayalı Öngörülü Kontrol Sistemi

M. Alphan Ulusoy

Elektronik Mühendisliği, Yüksek Lisans Tezi, 2009

Tez Danışmanları: Doç. Dr. Özgür GÜRBÜZ,
Yard. Doç. Dr. Ahmet ONAT

Anahtar Kelimeler: kablosuz ağ bağlantılı kontrol sistemleri, modele dayalı öngörülü kontrol, işbirlikli ortama erişim protokolleri

Özet

Ağ bağlantılı kontrol sistemleri dağıtık yapıları sayesinde geniş bir alana yayılmış veya yapısal olarak karmaşık tesislerin kontrolünde öne çıkarlar. Bu tür sistemler genellikle algılayıcıların, kontrollörlerin ve eyleyicilerin üzerinden haberleştikleri gerçek zamanlı kablolu ağlar kullanılarak gerçeklenirler. Son zamanlarda esneklik, kurulum kolaylılığı ve ölçeklenebilirlik gibi sebeplerden ötürü IEEE 802.11 v.b. kablosuz ağ teknolojileri, kontrol sistemi kullanımına adanmış kablolu ağların yerine kullanılabilecek bir seçenek olarak görülmeye başlanmıştır. Fakat, belirsiz gecikmeler ve veri kayıplarının sistem kararlılığına ve dinamiklerine olan olumsuz etkilerinden dolayı basit ağ bağlantılı kontrol sistemleri bu tür genel amaçlı kablosuz ağlar üzerinde çalışamaz.

Kablosuz ağ bağlantılı kontrol sorununa iki açıdan yaklaşan bu tezde belirsiz gecikmeler, peş peşe paket kayıpları ve kablosuz ağ trafiği altında dahi iyi bir kontrollör başarımı elde etmek için ortaklaşa çalışan bir kablosuz ağ bağlantılı kontrol sistemi ile bir işbirlikli ortama erişim protokolü tanıtılacaktır. Önerilen sistem özel bir test platformu üzerinde gerçeklenmiş, çok sayıda gerçekçi senaryo altında test edilmiş ve basit ağ bağlantılı kontrol sistemlerinin çalışamadığı peş peşe paket kaybı ve kablosuz trafik düzeylerinde çalışabildiği gösterilmiştir.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Networked Control Systems (NCS) where the components of a distributed feedback control system communicate over a network are typically made of three kinds of nodes. In an NCS, sensor nodes are responsible for periodically measuring plant outputs and communicating this data to controller nodes over the network. Controller nodes use plant outputs to calculate the control signals and communicate them to actuator nodes. Ultimately, actuator nodes apply the control signals to the plant.

NCS's are very appealing solutions to scenarios where the nodes of a control system have to be distributed spatially by design. Nevertheless, NCS's are *delay-sensitive* systems and thus total latency of the feedback loop must be bounded to ensure proper operation. Without loss of generality, end to end latency of an NCS can be broken down to 5 main components (Fig. 1.1): internal latencies of the sensor (τ_s), controller (τ_c), actuator nodes (τ_a), sensor to controller communication latency (τ_{sc}) and controller to actuator communication latency (τ_{ca}). Out of these delay components, τ_s, τ_c and τ_a refer to all computational and functional latencies that a particular node introduces and are bounded for hard real-time systems. Remaining delay components τ_{sc} and τ_{ca} refer to the latencies induced by the communication medium of choice and are not bounded for regular networks where medium access is based on random back-off times and state of the channel may require multiple retransmissions. Thus, basic NCS's (b-NCS) require dedicated real-time networks since they can not perform satisfactorily over a regular network due to the deteriorating effect of unbounded delays and packet losses. However, the overhead of installing a dedicated network often hinders the commissioning of the control system

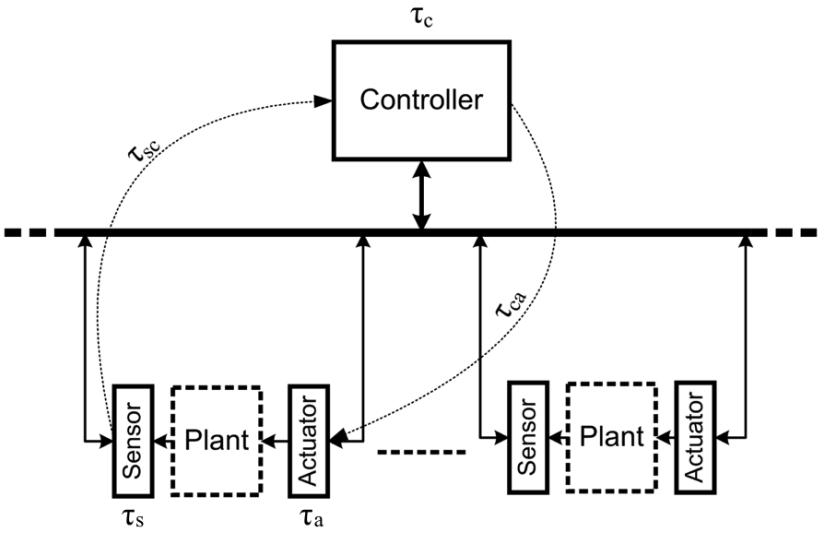

Figure 1.1: Latency components of a typical NCS.

and discourages its use.

In order to overcome the problem of unpredictable delays and loss that the data packets of an NCS are subject to when operating over a regular network, in [3] authors analyze the effects of the network on the control system and propose to take the characteristics of the network into consideration during the design of the control system. Similarly in [4] authors propose to compensate the disturbance force from transmission time delay with the disturbance observer included in the model of the control system. Despite the performance improvements reported in these works, making the communication medium an integral part of the actual controller being designed may not be a good design practice since the underlying network and the control system operating on it are two distinct entities and the traffic load and the delay of the communication medium can change during operation of the NCS. On the other hand, model predictive controllers are used in similar scenarios as given in Liu et al. [5] and [6] but these works either do not take the synchronization between the nodes into account or are not set up to be networked control systems due to the fact that they rely on a direct-link between the sensor and controller and a transmission failure would inhibit future predictions.

As a remedy to addressed problems, Model Based Predictive Networked Control System (MBPNCS) proposed in [7] improves the performance of a b-NCS under vari-

able time delays and packet losses by assuming standard NCS architecture with no requirement of direct links and a priori knowledge of the reference signal. MBPNCS, which operates over an ethernet LAN, employs a model based predictive controller which utilizes a model of the plant to predict control signals into the future and is shown to provide resilience against packet losses. However, the level of immunity MBPNCS provides against packet losses is only tested with a uniform packet loss model where probability of losing a packet during transmission is determined by a pseudo-random number generator with a uniform distribution which is not representative of true channel characteristics since packet losses are generally correlated and largely occur in bursts. Additionally, no experiments have been performed regarding the extent to which the traffic generated by other nodes on the network degrades the performance of the system.

Nevertheless, a truly flexible NCS requires wireless communication since it may well be the case that the nodes of the NCS have to be placed such that dedicated cabling for communication is not preferred or simply is not an option. However, when the nodes of the control system are distributed and the communication medium is air, transmission failures and delays owing to re-transmission attempts are no longer inconsequential as they are in the case of a properly designed field bus or some other dedicated, reliable and guarded wired communication medium. Thus, packet losses and unbounded delays occurring during wireless communication have to be rigorously evaluated and various counter-measures have to be taken during the design phase of a wireless NCS (WNCS).

In an attempt at improving the medium access latency of a wireless network so that a WNCS can operate on, several polling and time division multiple access (TDMA) based medium access control (MAC) protocols are presented in [8–12] that aim to minimize latencies caused by retransmission attempts due to collisions between wireless stations. However, these works suggest no improvements on the quality of the wireless link and thus they cannot be directly utilized in a WNCS due to unbounded latencies caused by bursts of errors which are typical of wireless communication.

As the quality of the wireless link directly affects the latency of the packets of a WNCS, it must be addressed before focusing on any higher levels of the commu-

nication stack. In the past two decades wireless communications technologies have evolved to a point where traditional network architectures have given way to relaxed topologies in which any node is allowed to assist in the transmission of information. In contrast with the traditional idea of communicating on a point-to-point basis under the supervision of a central base station, in contemporary ad-hoc and sensor networks a source depends on the cooperation of other nodes for successful transmission of information to the desired destination. The necessity for cooperation along with the transformation in the network architectures has roused new ideas that aim to utilize the broadcast nature of wireless communication, long considered as a waste of energy and source of interference, for improved performance through cooperation.

Forming independent multi-hop wireless links reminiscent of multiple input multiple output (MIMO) schemes, cooperative communications employ nodes of a wireless network as a set of distributed antennas to gain higher link reliability, diversity, reduced fading and higher efficiency. Performance gains achievable with cooperation can be better appreciated when the bursty nature of the wireless channel is considered: when source (S) cannot communicate directly to destination (D) all by itself due to severe fading, using an error control mechanism such as Automatic Repeat Request (ARQ) will not be of much use since the $S-D$ link will remain in fade for some time. On the other hand, if a nearby relay (R) helps S by sending the overheard packet over the $R-D$ link which is independent from the $S-D$ link, the chances of successful transmission will be higher — especially after some processing at D.

Cooperative communications have its origins in relay channels which are initially studied in [13]. Subsequently, this matter has been approached from various perspectives: as [14–18] focus on the physical (PHY) layer; [19–24] concentrate on the medium access control (MAC) layer and above along with some cross-layer issues. Among these, the Cooperative MAC (COMAC) protocol introduced in [23] stands out with its IEEE 802.11 compatibility, frame formats and frame exchange procedures that can be derived from IEEE 802.11 and lack of need for a "cooperation table" which is quite expensive to form and maintain. Nevertheless, most of these works emphasize that cooperation between the nodes either results in higher

throughput or reduced power consumption. Although [12, 25] examine the effects of cooperative communications on the reliability and latency of data packets from the perspective of wireless sensor networks (WSNs), little work has been done on this issue from the perspective of low-latency high-performance cooperative networks. This can be attributed to the limited focus of previous research which has been confined to either wireless sensor networks from the perspective of energy consumption or ad-hoc multimedia communication from the perspective of throughput. However, cooperative communications can also lower the latency of data packets in a fading channel by decreasing the number of required re-transmissions owing to improved probability of successful transmission. Thus, cooperative communication schemes can make a given wireless channel more suitable for delay-sensitive applications.

In this work, a novel Wireless MBPNCS (W-MBPNCS) [26] and a faithful implementation of the COMAC protocol are presented which work jointly to solve the problem of operating an NCS over a wireless network.

W-MBPNCS overcomes the limitations of traditional cabled NCS's such as lack of mobility and need for dedicated infrastructure by operating over an 802.11b wireless ad-hoc network formed between its nodes. High medium access latencies induced by ambient wireless traffic is handled with a modification of 802.11b medium access control (MAC) parameters, giving W-MBPNCS a greater priority in the presence of other nodes contending to access the medium. Unbounded packet latency is reduced to packet loss by introducing relative packet deadlines at each node of the system after which packets are assumed to be lost. The effect of packet loss is minimized by the estimations of the model based predictive controller. The implemented W-MBPNCS is thoroughly tested in demanding realistic scenarios and is shown to be resilient against packet losses, delays and ambient wireless traffic.

On the other hand, COMAC protocol utilizes the neighboring nodes in a wireless network to cooperate with each other for transmission of data packets. COMAC decreases receive thresholds through diversity receivers at the receiving nodes, providing longer transmission ranges and higher packet success rates for a given transmission power compared to non-cooperative communication. The implemented COMAC protocol is thoroughly tested in different scenarios and is shown to improve the quality of the wireless link significantly, resulting in superior controller performance

when the nodes of the W-MBPNCS communicate over a Rayleigh fading channel.

Chapter 2

Background

In this chapter, an overview of the overall architecture is presented followed by the details of wireless access, channel models, maximal ratio combining and the control algorithm used to evaluate the performance of the system.

2.1 System Architecture

W-MBPNCS is made up of 4 components as given in Fig. 2.1: the sensor node, the controller node which also contains the model $\hat{P}$ of the plant, the actuator node and the actual plant P. During the operation of W-MBPNCS, the sensor periodically reads plant outputs and communicates this data to the controller over the ad-hoc wireless network. In addition to calculating the control signal, the controller also predicts an additional number of control signals into the future using $\hat{P}$. Upon retrieval of controller packets, the actuator applies appropriate control signals to the plant. The functionalities of the nodes of W-MBPNCS are discussed in greater detail in Chapter 3.

A typical cooperation scenario using COMAC consists of two stages involving three nodes as depicted in Fig. 2.2. In *stage 1* the source node (S) disseminates a packet to the destination node (D) which is also overheard by the relay node (R). In *stage 2*, R cooperates with S in transmission of the packet to D. Diversity receiver at D combines the copies of the frame transmitted by S and R significantly increasing chances of successful reception due to improved SNR. Actual frame exchanges and frame formats of the COMAC protocol are presented in Chapter 4.

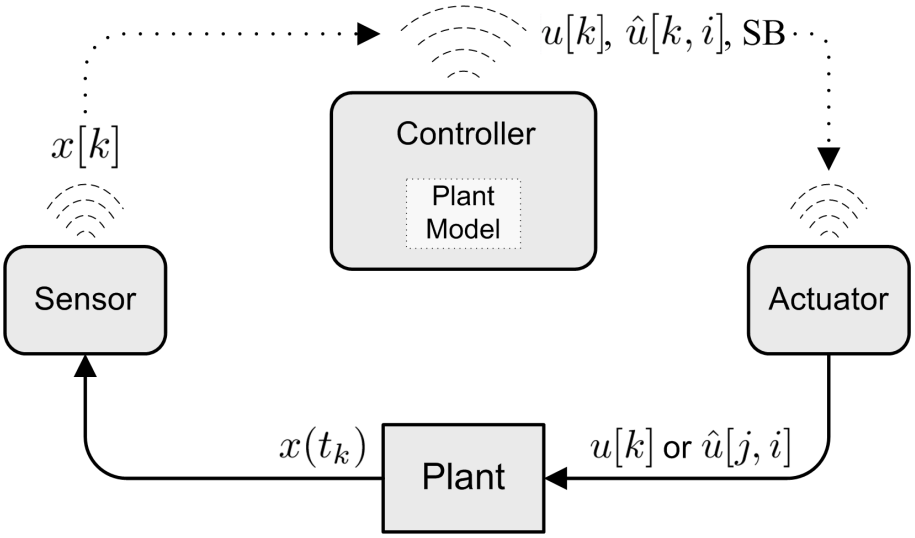

Figure 2.1: Overall architecture of W-MBPNCS.

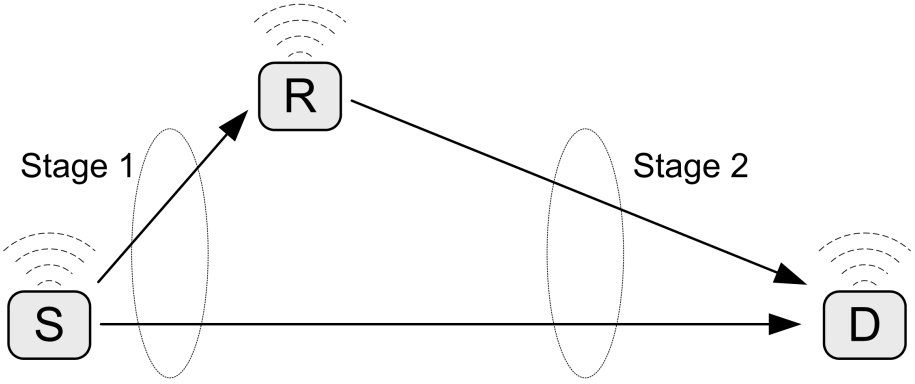

Figure 2.2: A typical cooperation scenario using COMAC.

2.2 Wireless Access

This section provides detailed information on the IEEE 802.11 standard; its MAC frame formats, basic access method and RTS-CTS protection mechanism. For the sake of brevity, only those details that are of primary importance to this thesis are covered here; the interested reader is kindly referred to [1] for further information.

2.2.1 MAC Frame Formats

All wireless stations (STAs) employing the IEEE 802.11 standard, exchange information using well defined frames each consisting of three components:

1. A *MAC header* which consists of several subfields such as: frame control, duration, address, sequence control and QoS control information;
2. A *frame body* of variable length which contains the actual payload;
3. A *frame check sequence (FCS)* which is the IEEE 32-bit CRC.

Fig. 2.3 depicts the general MAC frame format which contains a series of fields that occur in the same order in all frames. Out of these fields, *frame control, duration/id, address 1* and *FCS* form the minimal frame format and exist in all types of frames whereas rest of the fields are present only in specific frames.

Bytes:	2	2	6	6	6	2	6	2	0-23424	4
Frame Field:	Frame Control	Duration / ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	QoS Control	Frame Body	FCS

Figure 2.3: General MAC Frame Format of IEEE 802.11

- *Frame Control*: Individual subfields of the *frame control* field given in Fig. 2.4 have the following meanings:
  - *Protocol Version*: This field is two bits wide and its value is fixed to "00" for the IEEE 802.11 standard.
  - *Type and Subtype*: Type and subtype fields jointly determine the actual purpose of the frame. Valid combinations of these two fields can be found in [1].

- *To DS and From DS*: When considered as a two bit wide field, "00" means a direct frame from a STA to another STA, "01" means a frame from a STA to an AP, "10" means a frame from an AP to a STA and "11" is unused.
- *More Fragments*: Means that a frame will be followed by another fragment when set to "1".
- *Retry*: This field is set to "1" in any frame which is a retransmission of a previous frame to help the receiving STA with the duplicate detection and recovery.
- *Power Management*: Indicates the power management mode of a STA.
- *More Data*: Used in power saving mode.
- *Protected Frame*: A value of "1" indicates that *frame body* is encrypted.
- *Order*: Set to "1" during transmission of fragmented frames indicating that the order of the individual fragments must be preserved.

Bits:	B0-B1	B2-B3	B4-B7	B8	B9	B10	B11	B12	B13	B14	B15
Sub-Fields:	Protocol Version	Type	Subtype	To DS	From DS	More Frag	Retry	Pwr Mgt	More Data	Protected Frame	Order

Figure 2.4: Frame Control Field

- Duration/ID: This field contains a duration value in microseconds depending on the stage of the ongoing transmission and the type of frame being transmitted. This value is used to update the network allocation vector (NAV) of all receiving stations except the actual recipient of the frame.
- Address Fields: There are four address fields in the general MAC frame format Fig. 2.3) and each address field contains a 48-bit MAC address. In an ad-hoc IEEE 802.11 network individual address fields have the following meanings:
  - *Address 1*: Contains the receiver address (RA) identifying the final recipient of a frame.
  - *Address 2*: Contains the transmitter address (TA) identifying the station that a particular frame originates from.

- *Address 3*: Contains the unique address (BSSID) of the independent basic service set (IBSS) that identify the ad-hoc network that a frame belongs to.
- *Address 4*: Unused in IEEE 802.11 standard.

- Sequence Control: Sequence control field is made of two subfields as given in Fig. 2.5. *Fragment number* subfield is 4 bits wide and contains the sequence numbers of individual fragments of a fragmented transmission. *Sequence number* is a 12 bits wide subfield indicating the sequence number of a frame to be used in duplicate detection, frame ordering, etc.

Bits:	B0-B3	B4-B15
Subfields:	Fragment Number	Sequence Number

Figure 2.5: Sequence Control Field

- QoS Control: This 16-bit wide field contains the Quality-of-Service (QoS) related information of a MAC frame and exists in all frames transmitted between QoS STAs. Bits 5-6 determine the ACK policy at the receiving STA and a value of "01" prevents any ACK frames to be sent by the recipient STA.
- Frame Body: This is a variable length field that contains the actual payload of a frame.
- FCS: 32-bit CRC calculated over all the fields of the *MAC header* and the *frame body* field.

Request to Send Frame Format

Request to send (RTS) frame (Fig. 2.6) is sent whenever a transmitting STA wants to use NAV protection for its traffic destined for another STA. *Duration/ID* field is set to the duration in microseconds that the pending transmission is expected to take: e.g. time required to transmit RTS, CTS, DATA, ACK frames and three SIFS intervals in-between. RA and TA are the MAC addresses of the receiving and transmitting STA respectively.

Bytes:	2	2	6	6	4
Fields:	Frame Control	Duration / ID	RA	TA	FCS

Figure 2.6: Request to send (RTS) frame format

Clear to Send Frame Format

Clear to send (CTS) frames (Fig.2.7) are transmitted either in response to an RTS frame as in RTS-CTS protection mechanism or spontaneously whenever a STA desires to distribute NAV information and protect its transmission from collisions using cts-to-self mechanism. In the former case, duration is the duration value of the immediately previous RTS frame minus time required transmit an RTS frame and one SIFS interval; whereas RA field is copied from TA field of the immediately previous RTS frame. In the latter case, the mechanism is called CTS-to-self and duration equals one CTS time plus DATA frame time plus ACK frame time plus 2 SIFS intervals and RA is the MAC address of the STA transmitting the CTS frame.

Bytes:	2	2	6	4
Fields:	Frame Control	Duration / ID	RA	FCS

Figure 2.7: Clear to send (CTS) and Acknowledgement (ACK) frame formats

Acknowledgement Frame Format

Acknowledgement (ACK) frame (Fig.2.7) is transmitted by a STA whenever a transmission that requires positive acknowledgement is successfully received. Duration field equals to zero if the immediately previous frame is the last frame of an exchange or is calculated from the duration field of the previous frame in case of fragmented transmissions. RA is copied from the TA field of the immediately previous frame.

DATA Frame Format

DATA frames (Fig. 2.8) contain the actual payload of a frame exchange in their frame body fields. The duration field of a DATA frame is set to the time required

to transmit DATA, ACK frames and one SIFS time in-between if the payload to be transmitted is not fragmented and the transmitting STA uses individually addressed DATA/ACK frames. If not, total length of the fragmented payload and the mechanism in use (e.g. RTS-CTS) is also taken into account for duration calculation. RA, TA and BSSID address fields equal to MAC addresses of the receiving STA, transmitting STA and the IBSS respectively for an ad-hoc network. Sequence number is incremented after each successful transmission using a module-4096 counter and QoS control field is set according to the priority of the payload.

Bytes:	2	2	6	6	6	2	2	0-23424	4
Fields:	Frame Control	Duration / ID	RA	TA	BSSID	Sequence Control	QoS Control	Frame Body	FCS

Figure 2.8: Data frame format

2.2.2 Basic Access Mechanism

IEEE 802.11 MAC uses a contention based medium access mechanism called *distributed coordination function* (DCF). DCF is responsible for avoiding collisions and resolving them when they occur as multiple wireless stations (STA) try to transmit simultaneously. Fig.2.9 from [1] illustrates the basics of DCF.

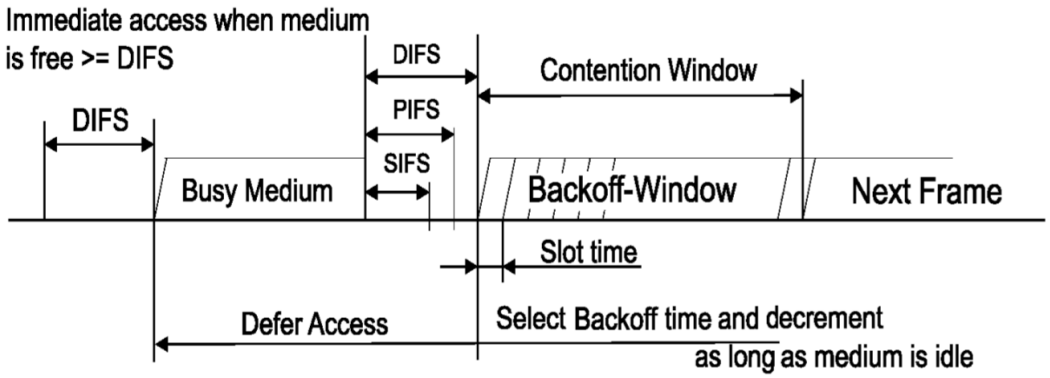

Figure 2.9: DCF Basic Access Method in IEEE 802.11b [1]

Functionality of DCF primarily depends on 5 key parameters: network allocation vector (NAV), DCF interframe space ($DIFS$), contention window (CW) and CW's minimum and maximum bounds (CW_{min} and CW_{max}). In a nutshell, DCF works

as follows: Each STA has a NAV which is updated by the duration value of a received frame whenever the STA is not the recipient of a received packet and the new duration value is bigger than the current value of the NAV. Each STA has a backoff timer which is loaded according to Eq. 2.1 whenever the medium is found to be busy. `Random()` is a pseudo-random integer from a uniform distribution over the interval $[0, CW]$. CW initially equals CW_{min} and is updated according to Eq. 2.2 before each retry until it reaches CW_{max}.

$$Backoff\_Time = Random() \times slot\_time \tag{2.1}$$

$$CW = 2^{retries} - 1 \tag{2.2}$$

Using DCF, a STA attempts to transmit only after reaching the end of its deferral period determined by the ongoing transmission as given in Fig. 2.10. Before each transmission, the STA check its NAV and if its NAV is zero senses the channel for an interval determined by the DIFS parameter. If the STA's NAV is non-zero or the medium is not free throughout DIFS, the STA defers transmission until the next time the channel is free; otherwise waits for an additional amount of time determined by its backoff timer. If the channel remains free until its backoff timer expires, the STA begins transmission; if not, it defers transmission until the next time the channel is free. Each successful transmission is concluded with an acknowledgment (ACK), thus absence of ACK indicates that the ongoing transmission is unsuccessful and a collision (contention) might have occurred. A STA whose packet is lost updates its CW parameter and reloads its backoff timer according to Eqs. 2.1, 2.2 before each retransmission. This scheme is repeated until all contentions are resolved and all STAs receive their ACKs.

2.2.3 Individually Addressed DATA/ACK Frames

If the payload to be transmitted does not require RTS-CTS protection, it is transmitted using a DATA frame after the success of basic access procedure as given in Fig. 2.11. If the receiving the STA can decode the DATA frame successfully, it sends an ACK frame immediately after one short interframe space (SIFS). Exchange is completed when this ACK frame is received by the original transmitter.

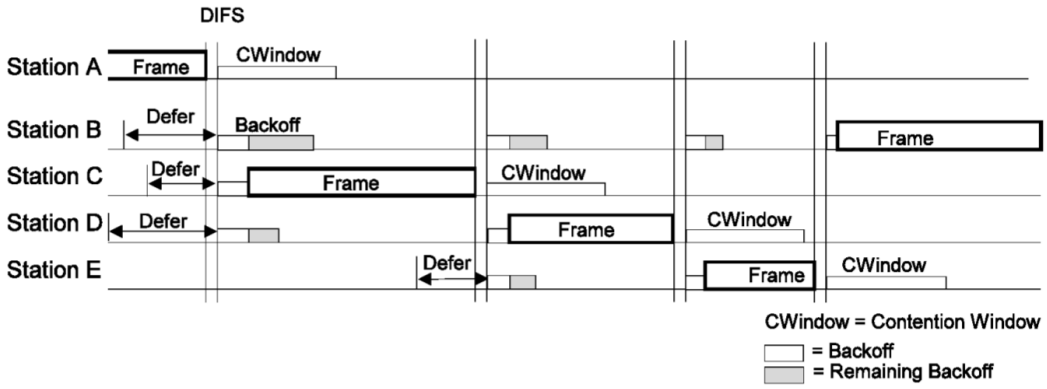

Figure 2.10: DCF Backoff Procedure in IEEE 802.11b [1]

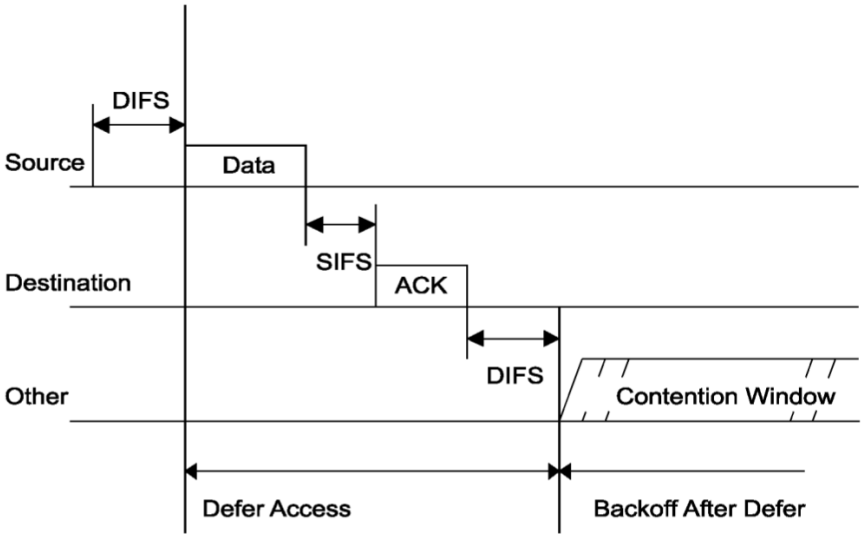

Figure 2.11: Transmission of DATA/ACK frames [1]

2.2.4 RTS-CTS Mechanism

RTS-CTS mechanism informs nearby STAs of the busy status of the channel for the duration of the transmission by distributing NAV information through the duration fields in the RTS and CTS frames. This mechanism can be used by a STA to reduce the chances of collisions during transmission by refraining nearby STAs from transmitting. A STA wishing to protect its transmission with the RTS-CTS mechanism sends an RTS frame to the destination STA before transmitting the DATA frame. The destination STA replies with a CTS frame. When the source STA receives the CTS reply, it disseminates the DATA frame which contains the actual payload and waits for the ACK frame. An ACK frame from the destination STA concludes the

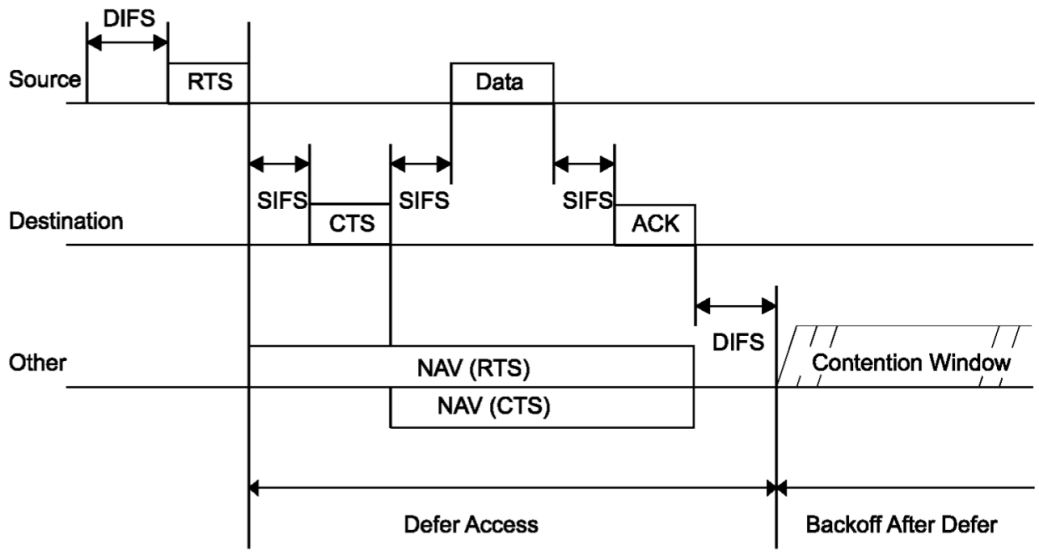

Figure 2.12: RTS-CTS Mechanism [1]

frame exchange.

2.3 Channel Models

Wireless communication is unreliable when compared to wired communication and suffers from mainly three issues as given in [27]: multipath fading, shadow fading and fast fading. *Multipath fading* is caused by the interference between the *out-of-phase* arriving copies of the same signal at the receiver. The phase difference results from the fact that these signals have travelled along different paths before reaching the receiver and thus have undergone different amounts of phase shifts. Since the phases of the arriving signals change rapidly either due to mobility of the obstacles around the wireless STA or due to mobility of the wireless STA, *multipath fading* causes rapid fluctuations in received signal strength (RSS) and intersymbol interference (ISI) between consecutive symbols resulting in irreducible error rates. *Shadow fading* occurs due to obstacles which block signals from arriving a wireless station and causes variations in the RSS for the same distance from the transmitter. *Fast fading* occurs when coherence time of the channel, duration in which characteristics of the channel remains correlated, is small relative to the delay constraint of the channel, duration in which the channel must preserve its characteristics for satisfactory operation. Since these issues are closely coupled to the surroundings of a

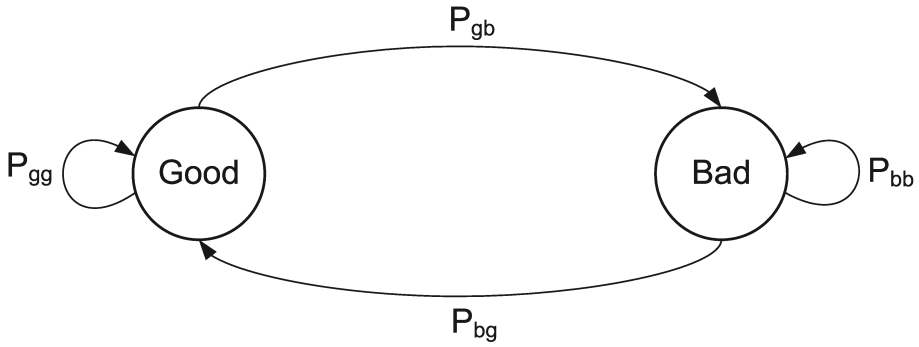

Figure 2.13: Gilbert/Elliot Model

wireless STA, characteristics of the wireless channel remain correlated for some time after a change. Thus, transmission errors and packet losses on the wireless channel occur in bursts followed by practically error-free periods rather than occurring completely randomly.

On the other hand, as the characteristics of a wireless channel depends on a multitude of factors, controlling all such contributing factors simultaneously to manipulate the characteristics of the wireless channel in a precise manner is very infeasible. Thus, instead of manipulating the surroundings of a wireless channel in an attempt to re-create the effects of a scenario, various channel models which readily encapsulate several important wireless channel characteristics under such scenarios are used to emulate their effects on the wireless channel in a controlled and reproducible way. In the following, three wireless channel models are introduced which are used to induce various channel impairments to an otherwise ideal wireless link in order to observe the effects of such impairments on the performance of the system.

2.3.1 Gilbert/Elliot Channel Model

In order to model the bursty packet loss characteristics of the wireless channel, this work uses a Gilbert/Elliot model [28]. The model is composed of two states, a *good* state and a *bad* state as shown in Fig.2.13 which determine the characteristics of the channel at a given time. In *good* and *bad* states of the channel, packets are lost according to packet loss probabilities P^g_{loss} and P^b_{loss} respectively. The next state of the channel is determined according to state transition probabilities P_{gb} and P_{bg} after each packet. Since state transition probabilities are typically small, the channel

state remains unchanged for some time after a transition is taken imitating bursts of packet loss when the model is in the *bad* state and periods of almost error free transmission when the model is in the *good* state.

Table 2.1: Summary of results of *longterm1* measurement from [2].

Fraction of Received Packets	93.5144%
Fraction of Lost Packets	6.4855%
Received Packets: Mean Burst Length	51.154
Lost Packets : Mean Burst Length	3.5457

In [2] measurement results of a wireless link in a realistic industrial setting using an IEEE 802.11b physical layer operating at 2 Mbps with QPSK is given. Tb. 2.1 summarizes the data used as the basis of the Gilbert/Elliot model. Interpreting the mean burst length of received and lost packets as average time spent in good and bad states (τ_g and τ_b), state transition probabilities P_{gb} and P_{bg} can be determined using Eqs. 2.3, 2.4.

$$P_{gb} = \frac{1}{\tau_g} \tag{2.3}$$

$$P_{bg} = \frac{1}{\tau_b} \tag{2.4}$$

Steady state probability of the model being in the good and bad states are given by Eqs. 2.5, 2.6.

$$P_g = \frac{P_{bg}}{P_{gb} + P_{bg}} \tag{2.5}$$

$$P_b = \frac{P_{gb}}{P_{gb} + P_{bg}} \tag{2.6}$$

Since this thesis uses the same mode of transmission as in [2] (2 Mbps with QPSK) when the Gilbert/Elliot model is used, model parameters (Tb.2.2) derived using above equations can be directly used here.

Table 2.2: Gilbert/Elliot Model Parameters

P_{gb}	P_{bg}	P^{g}_{loss}	P^{b}_{loss}
0.0196	0.282	0	1

2.3.2 Uniform Packet Loss Model

As a means for comparison, a much simpler *uniform packet loss model* is also implemented, where packets are dropped with a predetermined average probability with no correlation. A packet's fate is determined by comparing a random number to a predetermined threshold value. If the random number is greater, the packet is sent, if not it is dropped deliberately.

2.3.3 Rayleigh Fading Model

Both the Gilbert/Elliot and the uniform packet loss models are used solely for determining the fate of a packet and they provide no information about its signal strength. On the other hand, COMAC assumes that diversity receivers implement maximal ratio combining (MRC) whose functionality directly depends on SNR's of individual packets. Thus, a realistic channel model that can provide the signal strength of a received frame is required to be able to implement and test the performance of COMAC in a controlled fashion.

In practice, mobility of the transmitting and the receiving stations and the obstacles surrounding them cause rapid fluctuations in the received signal amplitude as out-of-phase copies of the same signal interfere constructively or destructively at the receiver after following different paths. Such a fluctuation is often modeled as a random variable with a particular distribution. For imitating the effects of multipath fading on the received signal amplitude observed in an environment reminiscent of a typical industrial setting with many obstacles and no direct line of sight between the transmitter and receiver Rayleigh distribution [27, 29] is the most widely used distribution whose probability density function (PDF) is:

$$f(x;\sigma) = \begin{cases} \frac{x}{\sigma^2} e^{\left(\frac{-x^2}{2\sigma^2}\right)} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{2.7}$$

and cumulative distribution function (CDF) is:

$$F(x;\sigma) = \begin{cases} 1 - e^{\left(\frac{-x^2}{2\sigma^2}\right)} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{2.8}$$

Since the power of a signal is proportional to the square of its amplitude, a change of variables $x^2 = P_{rx}$ reveals that Eq. 2.8 is essentially an exponential distribution with mean $2\sigma^2 = \bar{P}_{rx}$, where $\bar{P}_{rx}$ is the average received signal power:

$$F(P_{rx};\bar{P}_{rx}) = \begin{cases} 1 - e^{\left(\frac{-P_{rx}}{\bar{P}_{rx}}\right)} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{2.9}$$

Thus, this work uses an exponential random variable (Y) to implement the Rayleigh fading model whose output is passed to the MRC emulation in COMAC implementation. In order to incorporate path loss, this random variable is scaled by P_t/d^{α} where P_t is transmission power, d is the distance between the transmitting and receiving nodes and α is the path loss exponent.

2.4 Maximal Ratio Combining

Multipath fading causes serious degradation in received signal strength due to the interference between the multipath components of a signal. Since it is less likely for independent signal paths to suffer from deep fades simultaneously, diversity-combining of these independently fading signal paths in order to reduce the fading of the resulting signal is one of the most effective ways to mitigate the effects of multipath fading [29].

Maximal ratio combining (MRC) is a diversity combining technique where signals received at the branches an M-branch linear combiner (Fig. 2.14) are combined in

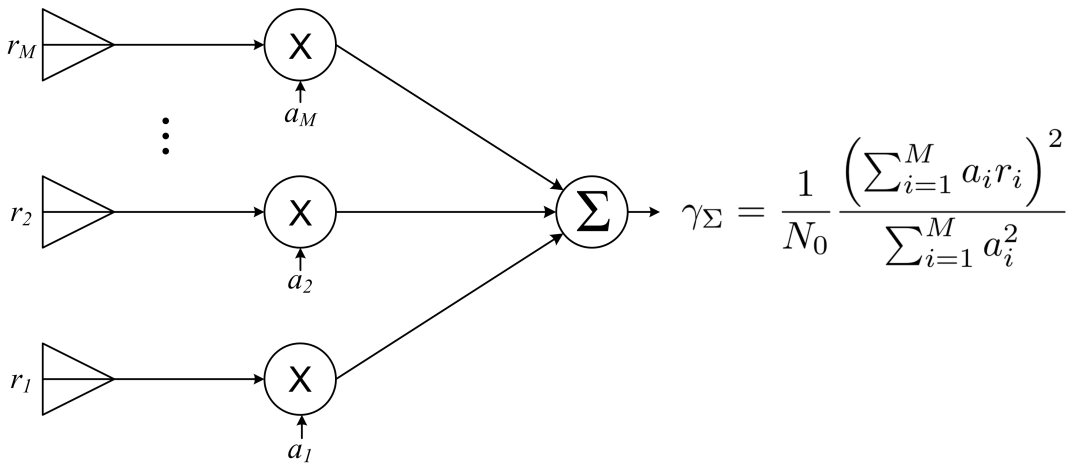

Figure 2.14: A linear combiner

such a way that output is a weighted sum of all branches. In an MRC combiner the weights of the individual branches are proportional to the signal to noise ratios (SNR) so that a branch with better reception is weighted more at the output. In an optimal MRC scheme weights of individual branches is $a_i = r_i/\sqrt{N_0}$ where a_i is the weight of an individual branch, r_i is the received signal envelope at an individual branch and N_0 is the noise power which is assumed to be equal at all branches. Thus, output of an optimal MRC becomes the sum of SNRs of each branch as given in Eq. 2.10:

$$\gamma_\Sigma = \sum_{i=1}^{M} \frac{r_i^2}{N_0} = \sum_{i=1}^{M} \gamma_i \tag{2.10}$$

where γ_Σ is the output SNR of the combiner and γ_i is the SNR of individual branches. This way, a diversity receiver with MRC reduces the impact of a faded independent signal path by utilizing the signal power received at other branches.

Assuming MRC at the receiver, COMAC protocol and its implementation achieves superior transmission at much longer ranges when compared to the non-cooperative scheme for a given transmit power.

2.5 The Plant and The Control Algorithm

There is a myriad of plants and control approaches that can be used to evaluate the performance of a discrete time control system. However, in order to ease the design task of the controller a linear approximation of the plant can be used as given in

Eq. 2.11 to employ a full state feedback approach as given in Eq. 2.12 where k is the sample index, u is the control signal, K is the controller gain matrix, r is the reference, G_r is the gain of reference and x is the plant state.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{2.11}$$

$$u[k] = G_r r[k] - Kx[k] \tag{2.12}$$

Additionally, if some of the state variables can not be measured directly, an observer can be used to estimate them as given in Eq. 2.13 where $\bar{A}$, $\bar{B}$, $\bar{C}$, $\bar{D}$ are discretized versions of A, B, C, D matrices, $\hat{x}$ is the estimated plant state and K_o is the observer gain matrix.

$$\begin{aligned} \hat{x}[k] = \bar{A}\hat{x}[k-1] + \bar{B}u[k-1] \\ + K_o(y[k] - \bar{C}(\bar{A}\hat{x}[k-1] + \bar{B}u[k-1])) \end{aligned} \tag{2.13}$$

Table 2.3: Parameters of the plant (Maxon RE-35 DC-motor)

J	$6.2800 * 10^{-6}$	$kg * m^2$
b	$2.1008 * 10^{-6}$	$N * m * s/rad$
Kt	$1.1854 * 10^{-1}$	$N * m/A$
Kv	$1.1789 * 10^{-1}$	$V * s/rad$
R	11.8000	Ω
L	$3.1613 * 10^{-3}$	H

To evaluate the performance of W-MBPNCS, this work aims position control of a Maxon RE-35 voltage controlled DC-motor (Tb. 2.3) with the following linear

approximation:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -b/J & K_t/J \\ 0 & -K_v/L & -R/L \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ 1/L \end{bmatrix}$$
$$C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, D = \begin{bmatrix} 0 \end{bmatrix}, x = \begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} \tag{2.14}$$

where b is the damping coefficient, J is the rotor moment of inertia, K_t is the torque constant, K_v is the speed constant, L is the terminal inductance, R is the terminal resistance, θ is the position, $\dot{\theta}$ is the speed and i is the current of the motor. Once the relevant parameters of the plant are obtained and a discrete-time model is prepared for 100 Hz sampling rate, a full state feedback controller for position control is implemented as given in Eq. 2.12.

In order to estimate the speed and the current of the motor an observer is used as given in Eq. 2.13. By selecting the observer gain K_o such that $\bar{C}K_o = I$ as given in [30], a reduced order Luenberger observer for the plant can be obtained as given in Eq. 2.15

$$\begin{aligned} x_1[k] &= y[k] \\ \hat{x}_2[k] &= \bar{A}_{21}x_1[k-1] + \bar{A}_{22}\hat{x}_2[k-1] + \bar{A}_{23}\hat{x}_3[k-1] + \bar{B}_2u[k-1] \\ \hat{x}_3[k] &= \bar{A}_{31}x_1[k-1] + \bar{A}_{32}\hat{x}_2[k-1] + \bar{A}_{33}\hat{x}_3[k-1] + \bar{B}_3u[k-1] \end{aligned} \tag{2.15}$$

where x_1 is the measured motor position (θ), x_2 is the estimated motor speed ($\hat{\dot{\theta}}$) and $\hat{x}_3$ is the estimated motor current ($\hat{i}$).

Simulation results show that the designed controller can follow a step reference at $2Hz$ with an error RMS percentage of 16.43% with respect to the applied reference signal. A time plot of plant output with respect to reference signal is presented in Fig. 2.15 and actual values of the discrete-time plant model and the controller matrices are given in Eqs. 2.16, 2.17.

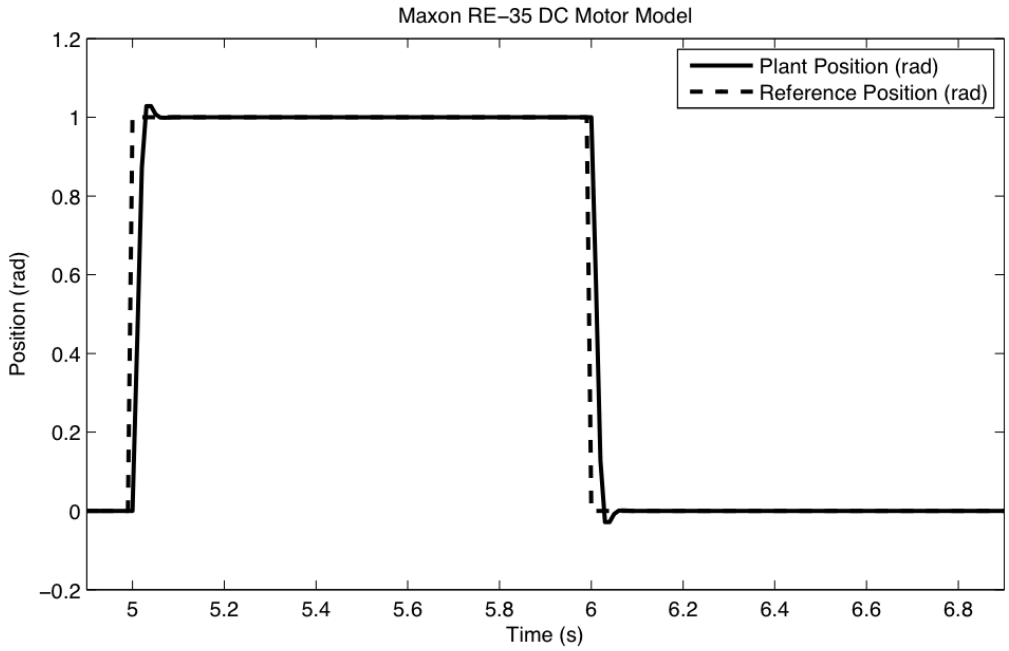

Figure 2.15: Plant output versus reference signal.

$$\bar{A} = \begin{bmatrix} 1 & 0.004571506466628 & 0.022911806165233 \\ 0 & 0.144036533192281 & 0.769533197614700 \\ 0 & -0.001520342229881 & -0.008122618558632 \end{bmatrix}$$
$$\bar{B} = \begin{bmatrix} 0.045961137637676 \\ 7.247567041901317 \\ 0.013024445382652 \end{bmatrix}, \bar{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, \bar{D} = \begin{bmatrix} 0 \end{bmatrix} \tag{2.16}$$

$$\bar{K} = \begin{bmatrix} 10.083697121434225 & 0.037594258027901 & 0.185128594546479 \end{bmatrix}$$
$$G_r = 10.083694665725522 \tag{2.17}$$

Chapter 3

Wireless Model Based Predictive Networked Control System (W-MBPNCS)

W-MBPNCS is a time-triggered discrete-time control system specially designed to operate in case of indeterministic bursty packet losses. This section focuses on the key points of the design of W-MBPNCS which are: per-node relative packet deadlines for reducing unbounded packet latency to packet loss, model based predictive controller and actuator state machine to compensate for the lost packets and increased medium access priority for more deterministic wireless channel access under high ambient wireless traffic.

3.1 Resilience Against Unbounded Packet Delays and Packet Losses

3.1.1 Per-node Relative Packet Deadlines

The sensor samples, appends a time stamp to and transmits the plant outputs at a period of $T = 1/f_s$ to the controller. The controller operates with a phase shift with respect to the sensor (typical network delay + $T/10$ in this case) introducing a relative deadline of $T/10$ for sensor packets. However, instead of starting $T/10$ after the 1^{st} sensor packet, the controller listens for the first n sensor packets ($n = 10$ in

this case) and compares the arrival times of the following $n-1$ packets with respect to their expected arrival times derived from the arrival time of the 1^{st} packet and the sampling time of the system. Since no packet can arrive earlier than expected, the most negative jitter in the arrival times of the following $n-1$ packets is used as an approximation of the additional latency of the 1^{st} packet. Using this information, the controller can initialize with the correct relative packet deadline even if the 1^{st} packet suffers an intermittent additional latency. The number of packets to wait for prior to initialization and the relative packet deadline depend on the quality of the wireless link and are determined empirically. Following initialization, the controller node checks the time stamps of the incoming packets and ignores any packets that fail to meet their deadlines, effectively reducing the unbounded packet latency to packet loss. The initialization and the time stamp mechanisms also work in the same way between the controller and the actuator.

3.1.2 Model Based Predictive Controller

As given by Onat et al. [7], in addition to calculating the control signal for the current time step, $u[k]$, the controller node also predicts n future control signal estimates $(\hat{u}[k,i], \{i:[1,n]\})$ using the model of the plant $\hat{P}$ iteratively which predicts future plant output estimates $(\hat{y}[k,i], \{i:[1,n]\})$ and state estimates $(\hat{x}[k,i], \{i:[1,n]\})$. The number of predictions n, which is 50 in this case, is chosen based on factors such as available bandwidth, characteristics of the wireless channel, characteristics of the plant, accuracy of the plant model and available processing power. Control signal estimate $\hat{u}[k,i]$ calculated at time step k is applied to the plant in case of a communication failure between the controller and the actuator i time steps later (at $k+i$). The source of the control packet produced at time step k can be either the incoming sensor data, $x[k]$, if it arrives on time for time step k, or the first state estimation produced at the previous time step, $\hat{x}[k-1,1]$. In the latter case, the control packet is valid only under the assumption that the previous control signal $u[k-1]$ which is applied to $\hat{P}$ at time $k-1$ is also applied to P implying that the controller packet sent at time step $k-1$ is not lost. Needless to say, this assumption is very unlikely to hold and thus a flag named *SB*, which stands for sensor based, is also stored in the controller packet indicating whether the control

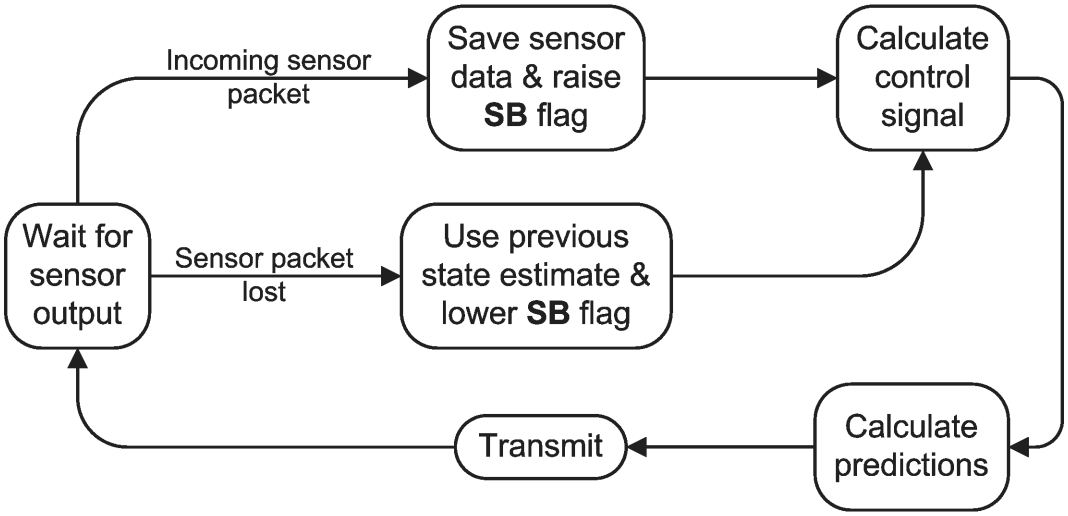

Figure 3.1: Operation of the controller node.

signals of a particular packet are based on sensor data or previous state estimations. The actuator node uses this *SB* flag to decide which control signal must be applied to the plant at a given time. A summary of the above discussion is illustrated in Fig. 3.1.

3.1.3 Actuator State Machine

Since the controller node calculates the control signal estimates of the plant model with the assumption that all of its packets are received by the actuator and all the control signals it calculates are applied to the plant, a scenario where:

1. At sampling time $[k]$ controller sends a sensor or model based packet to the actuator.

2. At sampling time $[k]$ actuator receives the packet and applies the control signal, $u[k]$.

3. At sampling time $[k+1]$ controller sends a *packet with* $r[k+1] \neq r[k]$ *or* $y[k+1] \neq \hat{y}[k,1]$ to the actuator.

4. At sampling time $[k+1]$ the packet is lost. Actuator applies the 1^{st} control signal estimate of the previous packet, $\hat{u}[k,1]$.

breaks this assumption since at step 4 $\hat{P}$ receives $u[k+1]$ calculated at step 3 whereas P receives the model based prediction $\hat{u}[k,1]$ calculated at step 1. Since $u[k+1] \neq \hat{u}[k,1]$ the states of the actual plant P and the model that runs inside controller $\hat{P}$ are no longer expected to be equivalent.

In summary, whenever a controller packet is lost, ignoring the exceptional cases where $r[k+1] = r[k]$ *and* $y[k] = \hat{y}[k,1]$, control signals sent by the controller become obsolete until the next time the controller is *synchronized* with the plant by receiving a sensor packet and successfully sending its *sensor based* calculations to the actuator. In order to cope with this problem, the actuator node embodies a state machine with two states similar to the actuator given by Onat et al. [7]: the *synchronized state* corresponding to the instants when states of $\hat{P}$ and P are synchronized and the *interrupted state* corresponding to the instants when states of $\hat{P}$ and P are out of synchronization.

Upon retrieval of a controller packet, the actuator node checks its current state and the SB flag of the incoming packet. If the actuator state machine is in the *synchronized state*, the control signal ($u[k]$) of each packet is applied to the plant regardless of the condition of the SB flag until a controller packet is lost and actuator state machine makes a transition to the *interrupted state*. In the *interrupted state*, incoming controller packets are ignored and predictions of the last packet received in the *synchronized state* are applied to the plant in a consecutive manner ($\hat{u}[j,i]$, { i : [1,n] } assuming that the transition to the *interrupted state* occurred at $j+1$) until a sensor based control packet is received indicated by a high SB flag. When such a packet is received, the actuator state machine makes a transition back to the *synchronized state*. If the actuator node runs out of predictions in the *interrupted state*, if i reaches n, it keeps applying the last control signal estimate $\hat{u}[j,n]$ to P until the actuator node makes a transition to the *synchronized state*. An overview of the functionality of the actuator is illustrated in Fig. 3.2.

3.1.4 Stability Considerations

When there is no packet loss between its nodes, W-MBPNCS acts as a regular discrete-time control system. However, during periods of packet loss P receives control signals based on $\hat{P}$'s state estimates instead of its own actual states. Thus,

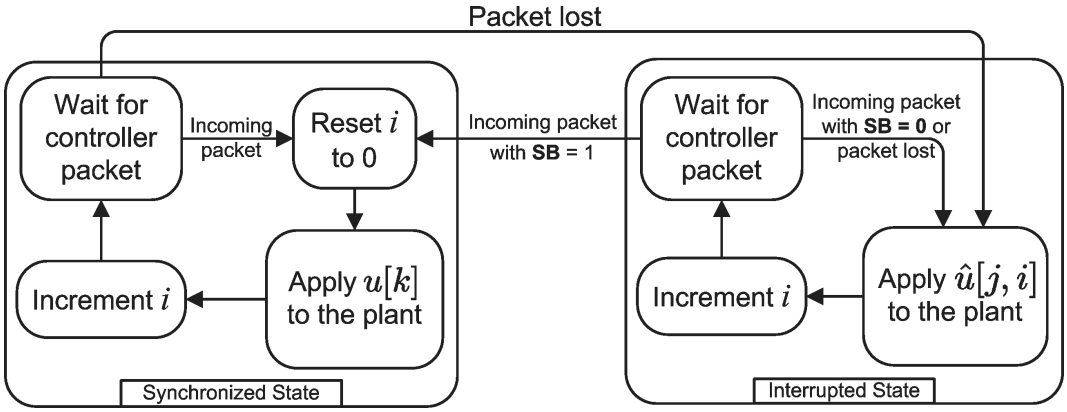

Figure 3.2: Operation of the actuator node.

at each consecutive packet loss, these control signal estimates deviate from actual control signals as $\hat{P}$'s state estimates deviate from P's actual states due to modelling errors. Consequently, during such intervals P's stability depends on the length of the packet loss burst and modelling errors in $\hat{P}$. An analysis of the stability conditions for MBPNCS during bursts of packet loss which is directly applicable to W-MBPNCS can be found in [31] and is not repeated here. Nevertheless, 50, as the number of predictions used in this work, is a suitable value for maintaining the stability of the W-MBPNCS platform in the experiments during bursts of packet loss.

3.2 Resilience Against Ambient Wireless Traffic

Wireless channel is of broadcast nature and thus, performance of a WNCS can be seriously affected by the wireless traffic created by neighboring STAs. When multiple STAs try to transmit data simultaneously, their transmissions interfere with each other causing their packets to be lost. Since STAs typically keep retrying until their packets are successfully sent, each collision increases the latency of a packet. However, since DCF is stochastic in nature with no upper bound on medium access latency, a packet can be delayed so long that by the time it is received by its destination the information it carries is no longer relevant. Thus, ambient wireless traffic effectively increases the packet loss rate of the channel since the control algorithm must receive sampled states of the plant at exact intervals and a

packet received after its expected time is equivalent to a lost packet.

As a remedy to this problem, similar to the ideas behind 802.11e [1], CW_{min} and CW_{max} values can be decreased in order to control the maximum delay spread in case of collisions. Following a collision, a STA with smaller CW_{min} and CW_{max} values clearly attempts to access the channel before any neighboring STAs with default values, increasing its chances for successful transmission since the neighboring STAs will defer their transmissions once they hear its transmission. Furthermore, a decreased $DIFS$ value also gives a higher priority to a STA in channel accesses as the STA with the smaller $DIFS$ value waits shorter and forces other STAs to defer their transmissions by starting its transmission sooner.

When there is a large number of neighboring STAs trying to transmit either because of being forced to defer their transmissions by mentioned MAC modifications or because of non-empty queues, chances of finding the medium busy at a given time will be higher. Nevertheless, these MAC modifications can prevent deadline misses even if a W-MBPNCS packet is delayed due to busy medium provided that sum of the remaining duration of the neighboring STA's ongoing frame exchange and the W-MBPNCS node's pending frame exchange is smaller than the relative packet deadline. If the remaining duration of the neighboring STA's ongoing frame exchange is long enough to violate this constraint, either due to low transmission rate or large packet payload, W-MBPNCS packets can miss their deadlines but mentioned MAC modifications are still highly beneficial due to the faster and more deterministic medium access they provide.

W-MBPNCS employs both of the suggested modifications to increase its nodes' medium access priority and to decrease the delay their packets are subject to, ultimately improving the overall performance of the system under ambient wireless traffic. When deployed on a factory floor or some other facility open to wireless traffic generated by other sources such as mobile devices and workstations; W-MBPNCS can operate with no major loss in performance owing to its modified 802.11b parameters.

Chapter 4

Cooperative Medium Access Control Protocol (COMAC)

COMAC [23] is a decode-and-forward cooperative MAC protocol that utilizes Maximal Ratio Combining (MRC) implemented in the diversity receivers of the cooperating nodes to translate decreased receive thresholds to longer transmission ranges and higher packet success rates compared to non-cooperative communication.

However, for COMAC to work, cooperating nodes must be able to make use of other nodes' overheard frames (e.g. R must be able to send a frame to D in response to an overheard S-to-D frame). On the contrary, the virtual carrier sensing mechanism implemented in IEEE 802.11 dictates that all STAs that overhear a frame which is not addressed to themselves must set their NAVs according to the duration information of the frame, drop the frame and remain silent for the entire transaction. COMAC modifies this behavior for cooperating nodes to let them exchange frames with each other in response to frames overheard from other COMAC nodes. Although this modification makes cooperative communication possible, it also raises the need for a mechanism to distinguish between a cooperative and non-cooperative exchange. To meet this end, COMAC uses custom frame formats which are distinguished from standard IEEE 802.11 frames through the modified reserved bits of their frame control fields.

For the sake of brevity, this work considers only the case where S always requests cooperation from the same R and that particular R is always available for cooperation as details of the mechanisms involved in actuating relays with respect

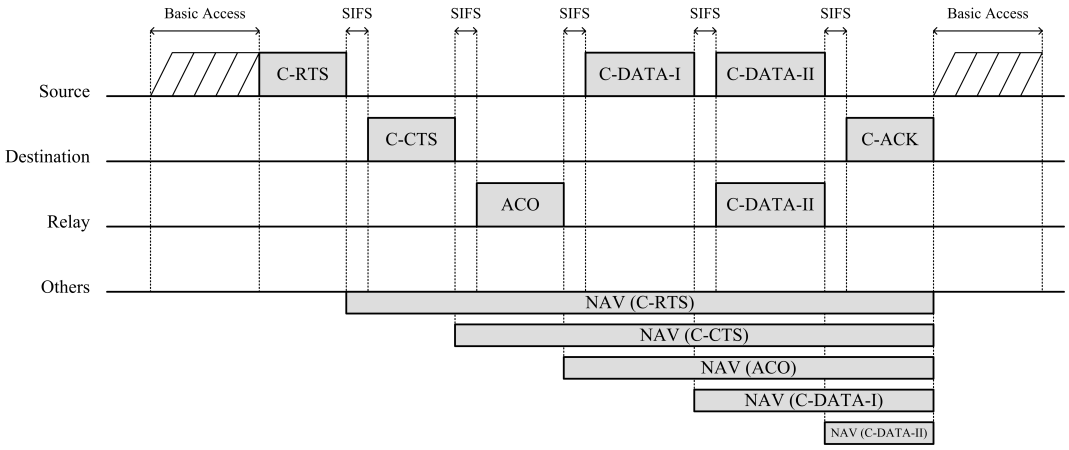

Figure 4.1: COMAC frame exchange and NAVs of neighboring STAs.

to some criteria is beyond the scope of this work. Interested reader is referred to [24] in which authors introduce an efficient distributed relay actuation mechanism for COMAC.

The rest of this chapter introduces frame formats used by COMAC and discusses the details of the protocol.

4.1 COMAC Frames

COMAC uses 5 special frames which are distinguished from regular IEEE 802.11 frames by their modified reserved frame control bits. A cooperative communication is initiated by a C-RTS frame sent from S to D reserving the medium for one COMAC exchange. C-RTS frame (Fig. 4.2) is a modified version of the regular RTS frame which includes an additional *Cooperation Information* field which holds the 6 byte MAC address of the node that S requests cooperation from. After receiving the C-RTS frame, D replies S with a C-CTS frame (Fig. 4.3). Overhearing the C-RTS and C-CTS frames, the relay which is available for cooperation sends an ACO (available to cooperate, Fig. 4.3) frame to S. Following this, the two stage cooperative communication commences. In stage-1 S disseminates the C-DATA-I frame (Fig. 4.4) to D which is also overheard and decoded by R. In stage-2 S and R send the C-DATA-II (Fig. 4.4) frame, which holds the same payload as C-DATA-I, simultaneously to D. After combining and successfully decoding the received data packets, D ends the transaction with a *C-ACK* (Fig. 4.3) frame. Fig. 4.1 illustrates

a successful COMAC transaction and NAVs of neighboring STAs.

Bytes:	2	2	6	6	$n * 6$	4
Fields:	Frame Control	Duration / ID	RA	TA	Cooperation Information	FCS

Figure 4.2: C-RTS frame format

Bytes:	2	2	6	4
Fields:	Frame Control	Duration / ID	RA	FCS

Figure 4.3: C-CTS/ACO/C-ACK frame formats

Bytes:	2	2	6	6	6	2	0-23424	4
Fields:	Frame Control	Duration / ID	RA	TA	BSSID	Sequence Control	Frame Body	FCS

Figure 4.4: C-DATA-I/C-DATA-II/DATA frame formats

4.2 Protocol Details

COMAC protocol defines three roles corresponding to three types of nodes (S, R and D) each with a specific set of tasks to be carried out during a COMAC frame exchange. In the following, each node's tasks are discussed in detail.

4.2.1 Source

The source node determines whether the communication that is about to start is cooperative or not. If the source chooses not to cooperate, then IEEE 802.11 standard frame exchanges (RTS/CTS mechanism or individually addressed data frames) are utilized.

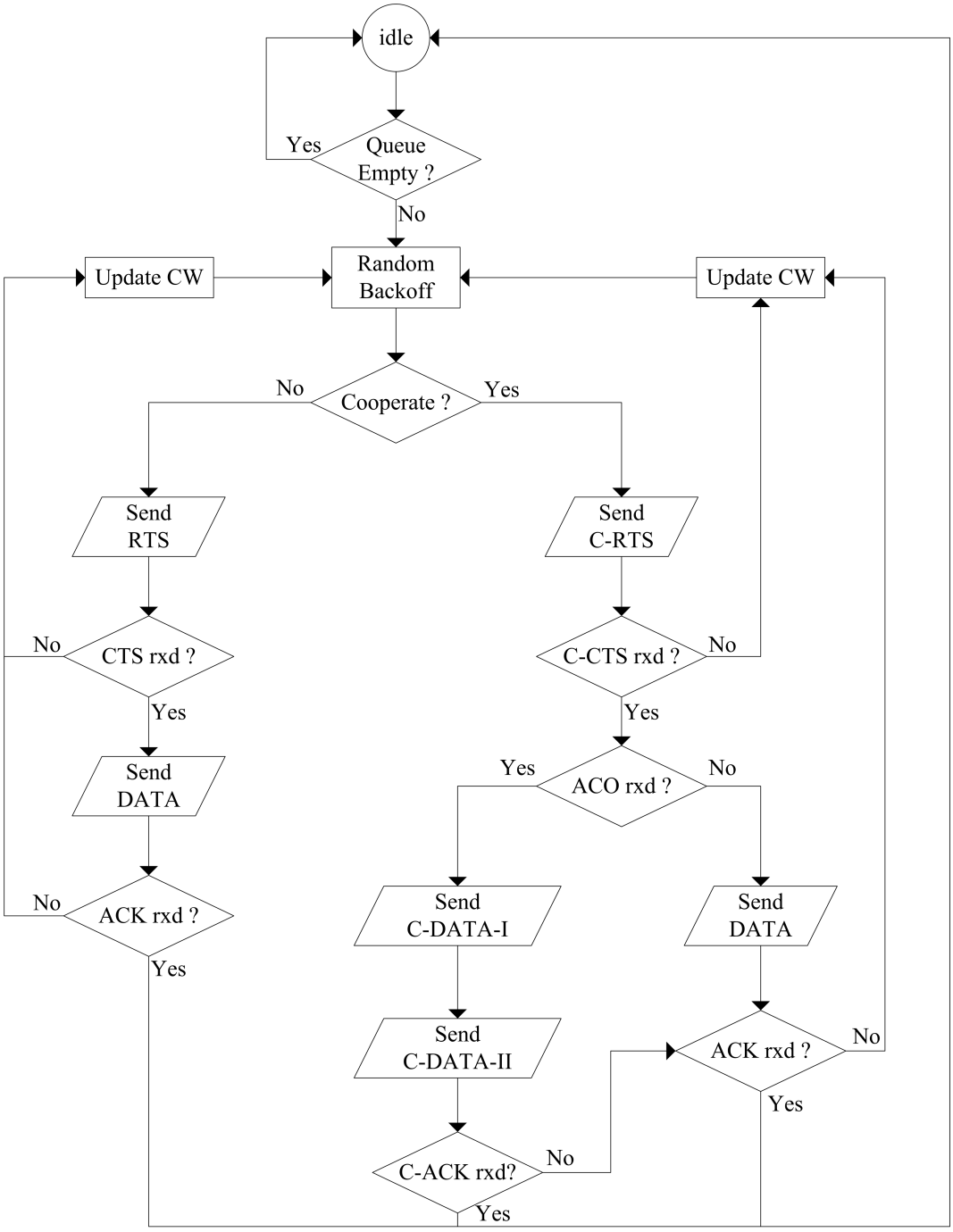

Figure 4.5: Operation of the source node.

Provided that S chooses to cooperate, it sends a C-RTS frame to D requesting cooperation from the R node whose MAC address is stored in the *cooperation information* field and reserving the medium by informing neighboring STAs about the duration of the cooperative communication stored in the *duration* field. Duration information of the C-RTS frames is given by Eq. 4.1:

$$D_{C-RTS} = T_{C-CTS} + T_{ACO} + 2 * T_{C_DATA} + T_{C-ACK} + 5 * T_{SIFS} + 6 * T_{prop} \quad (4.1)$$

where T_{SIFS} is the duration of one short interframe spacing (SIFS), T_{prop} is the time it takes for a signal to cross the maximum allowed distance between two STAs and $T_{C-CTS}, T_{ACO}, T_{C_DATA}, T_{C-ACK}$ stand for the durations of C-CTS, ACO, C-DATA and C-ACK frames respectively.

If a C-CTS frame is received in response to the C-RTS frame, S waits for the ACO frame from R. If C-CTS is not received after 1 SIFS interval, S updates its contention window and starts over.

After receiving the ACO frame S knows that R is available to cooperate, and it can start cooperative communication. S sends the C-DATA-I frame to D which is followed by the C-DATA-II frame after 1 SIFS time and waits for the C-ACK frame. If an ACO frame is not received after the C-CTS frame, S concludes that R is not available and reverts back to non-cooperative mode communication. In this case, S sends a DATA frame and waits for an ACK frame. Apart from their different frame identification bits, the only difference between C-DATA-I, C-DATA-II and DATA frames is their duration information which is set according to Eq. 4.2.

Operation of the source node (S) is given in Fig. 4.5.

$$\begin{aligned} D_{C-DATA-I} &= 2 * T_{C_DATA} + T_{C-ACK} + 2 * T_{SIFS} + 3 * T_{prop} \\ D_{C-DATA-II} &= T_{C_DATA} + T_{C-ACK} + T_{SIFS} + 2 * T_{prop} \\ D_{DATA} &= T_{C_DATA} + T_{C-ACK} + T_{SIFS} + 2 * T_{prop} \end{aligned} \quad (4.2)$$

4.2.2 Destination

An incoming C-RTS frame means a request for cooperative transmission and the receiving node (D) replies with a C-CTS frame and starts waiting for an ACO frame from R.

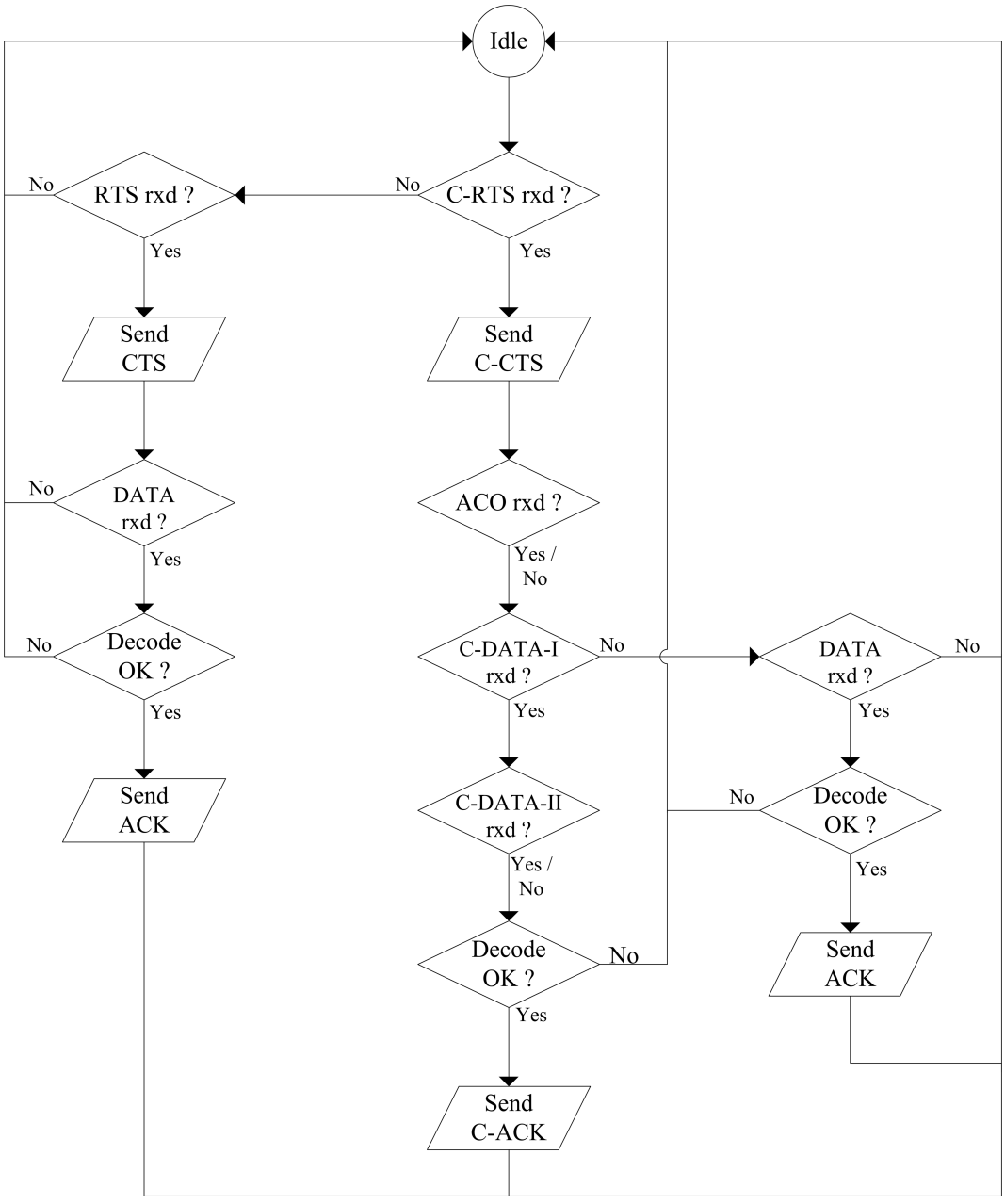

Figure 4.6: Operation of the destination node.

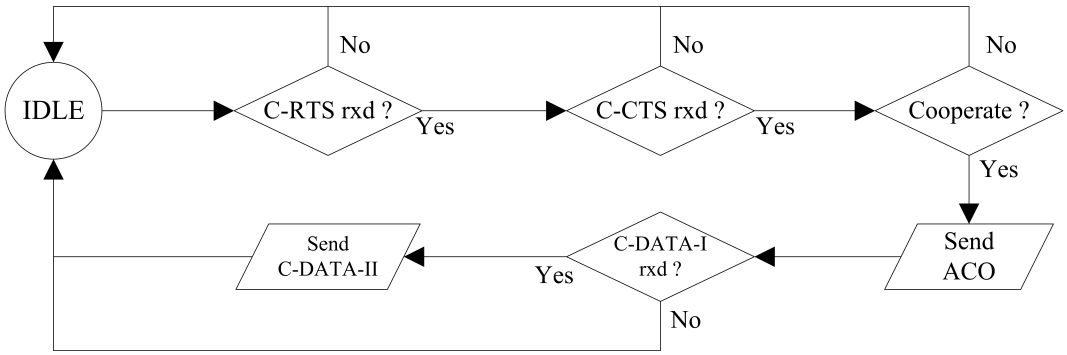

Figure 4.7: Operation of the relay node.

If an ACO frame is received, D concludes that R is available and waits for the C-DATA-I and C-DATA-II frames. Provided that both C-DATA-I and C-DATA-II frames are received and decoded successfully, D sends a C-ACK to S indicating that cooperative communication was successful.

If an ACO frame is not received, D assumes that S will transmit without cooperation so waits for a $DATA$ frame and replies with an ACK frame indicating that cooperation failed but transmission was successful. On the other hand, if D receives a C-DATA-I frame while waiting for a DATA frame, it assumes that cooperation has been initiated and operates as in the case above.

However even though cooperative communication is initiated cooperation may fail if:

- D does not receive any C-DATA-II frames and uses only the received C-DATA-I frame to recover the payload,
- D receives a DATA frame after ACO instead of C-DATA-I meaning that cooperation was aborted.

In these cases if D can recover the payload sends S an ACK frame meaning that payload was successfully received in spite of failed cooperation.

Operation of the destination node (D) is given in Fig. 4.6.

4.2.3 Relay

If a STA over-hears a C-RTS frame with its MAC address in the *cooperation information* field, it infers that assistance as a relay (R) is requested and waits for

the C-CTS frame. If a C-CTS frame is received, R decides to cooperate or not determining the mode of the transmission.

If R decides to cooperate, it sends a ACO frame to S one SIFS time after the reception of the C-CTS frame and starts waiting for the C-DATA-I frame. If a C-DATA-I frame is received, R repeats the received C-DATA-I frame in the form of a C-DATA-II frame one SIFS time later. In contrast with D, R does not expect an ACK or C-ACK frame but it remains silent until the end of the ongoing transaction to prevent possible collisions. If R decides not to cooperate, it returns to idle state and remains silent until the end of the ongoing transaction.

Similar to the case in the D node, cooperation may fail after being initiated if a DATA frame is received instead of a C-DATA-I frame indicating that cooperation was aborted.

Operation of the relay node (R) is given in Fig. 4.7.

Chapter 5

Implementation Details

5.1 Implementation Platform

5.1.1 Hardware

The platform on which W-MBPNCS and COMAC are realized and tested is, for the most part, made up of 4 hardware components (Figs. 5.1, 5.2):

- Advantech PCM-9584 PC104+ industrial computer board (2 GHz Intel Pentium M with 2 GB of RAM).
- CNET CWP-854 pci 802.11g wireless NIC.
- Mesa 4i30 quadrature counter daughter board.
- Kontron 104-ADIO12-8 ADC/DAC daughter board.

Choice of Wireless NIC

Although most quadrature counter and ADC/DAC boards have quite generic programming interfaces based on simple I/O accesses to several registers via the ISA bus, wireless NICs come with sophisticated software support (drivers) in the form of kernel modules which are responsible for interfacing various infrastructures of the operating system to the wireless NIC. Thus; stability, availability and code quality of the driver is of utmost importance since this thesis places considerable emphasis on wireless communication. A thorough research on the availability of open source drivers and development groups for different wireless chipsets revealed that there are

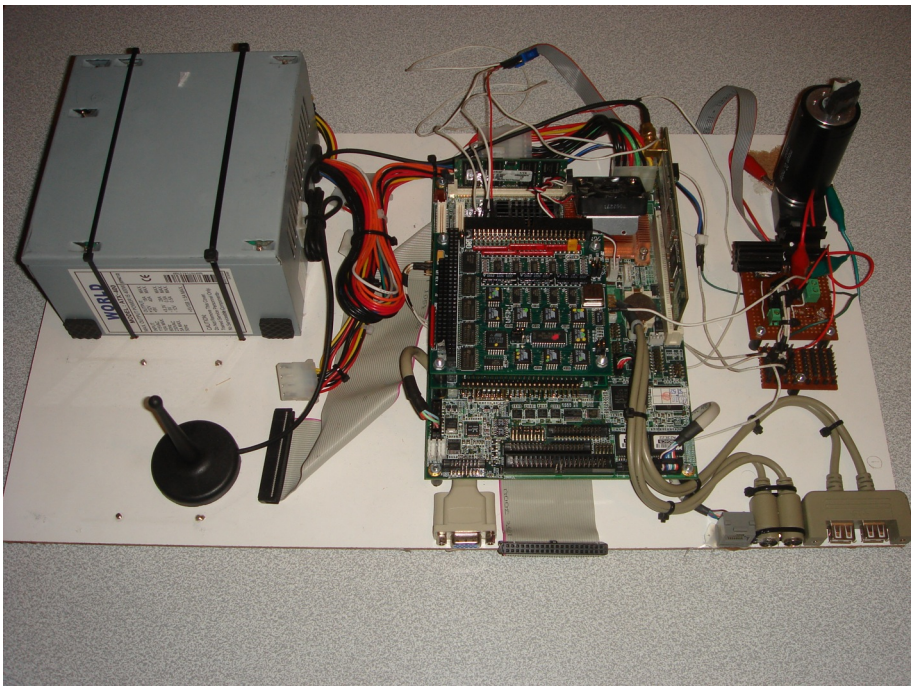

Figure 5.1: SENS/ACT node

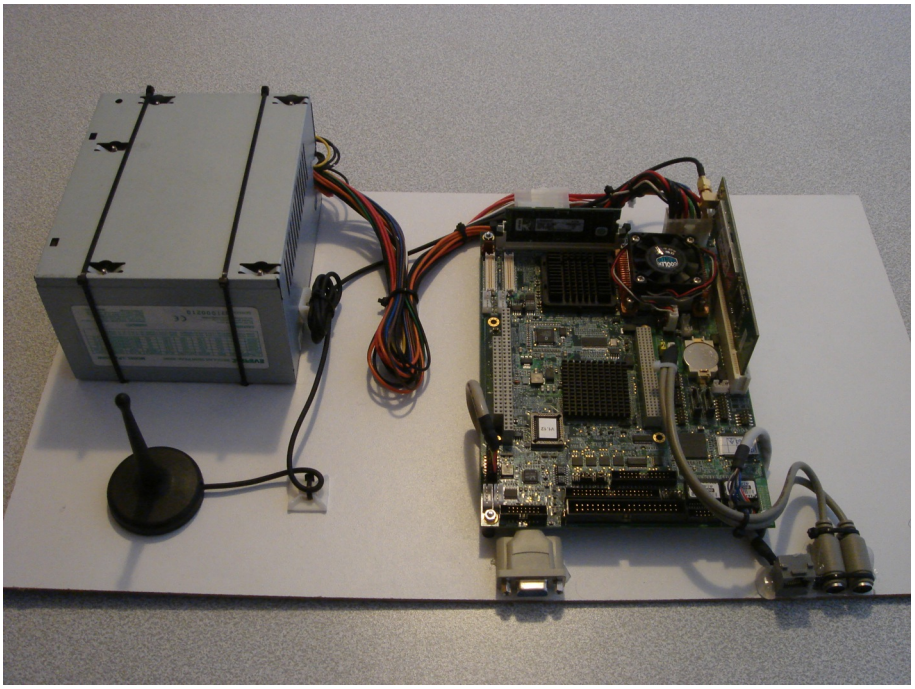

Figure 5.2: CTRL and TRA/RELAY nodes

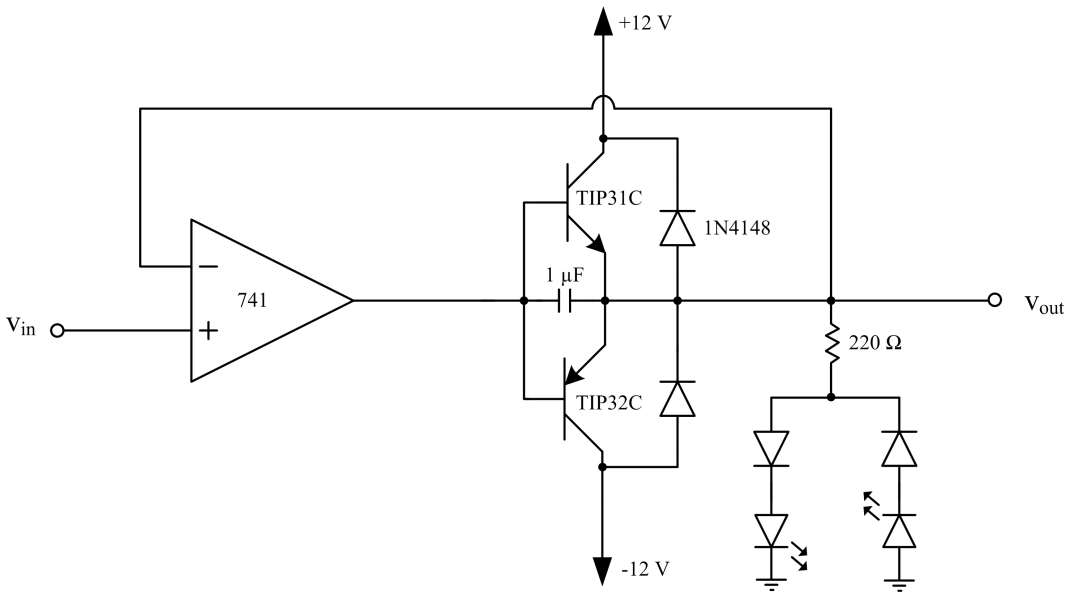

Figure 5.3: Schematic of motor driver circuit

active driver development groups for ralink, prism, ipw220 and zd1211rw chipsets. After a market research, the most feasible NIC was found to be CNET CWP-854 (PCI) 802.11g WLAN card with a RALINK 2561 chipset with open source drivers developed by the rt2x00 project[32]. This NIC was installed after a slight modification on its hardware: LEDs on the card were unsoldered and soldered on the back to make the card fit in the PCI slot.

Computer-Plant Interface

Maxon RE-35, the plant, is a voltage controlled DC-motor with a starting current of 4.16 A, a nominal current of 0.915 A and an encoder resolution of $\pi/1000$ radians (2000 counts per revolution). Since the maximum output current of the ADC board is rated 3 mA per channel, a voltage follower motor driver circuit (Fig. 5.3) with a peak output current of 5 A and an output current of 3 A is designed and implemented to be able to drive the plant. To measure the angular position of the plant, output of the encoder is connected to the quadrature counter. Fig. 5.4 illustrates the interface between the plant and the computer.

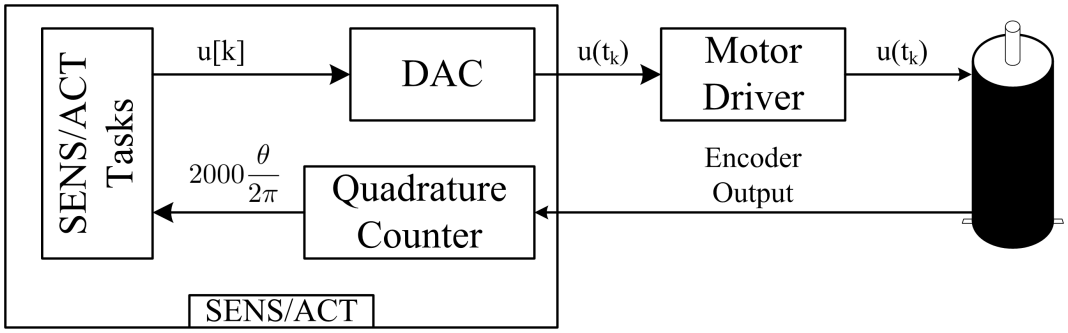

Figure 5.4: Plant-computer interface

5.1.2 Software

In order to be able to realize a time-triggered discrete-time control system on an advanced platform running a sophisticated operating system (OS) such as GNU/Linux, the OS must be real-time to satisfy the timing requirements of the tasks such that all periodic tasks of the control system (e.g. sampling of plant outputs, calculation of plant inputs, application of these inputs etc.) are executed accurately at predetermined points in time dictated by the sampling rate of the control system. There are several real-time extensions that provide real-time functionality on GNU/Linux. These extensions usually consist of a kernel patch, support libraries and application programming interfaces (APIs). The kernel patch typically introduces a real-time scheduler which is responsible for scheduling and executing real-time tasks of the system. This real-time scheduler runs the GNU/Linux (the whole operating system itself) as the lowest priority task whereas the rest of the tasks (non real-time, OS dependent) are scheduled and executed by the Linux scheduler — the lowest priority (background) task of the real-time scheduler.

Initially a platform based on a Linux 2.4.22 kernel patched with RTLinux 3.2-pre3 was prepared. However, this platform was abandoned due to RTLinux's lack of support for the current 2.6.x series kernels required by the test platform hardware. After some research, it was concluded that RTAI and Xenomai were two possible real-time framework candidates due to their active and rich development communities, stability and promising future. Since, their performances are similar as stated in [33]; the decision was totally based on documentation quality and manageability in which Xenomai is slightly better. Current real-time GNU/Linux platform is

based on a Debian 4.0 GNU/Linux distribution and a custom Linux 2.6.27.10 kernel with Adeos I-pipe and Xenomai 2.4.6.1 patches. Adeos I-pipe patch introduces the interrupt pipeline functionality on which Xenomai depends to provide real-time functionality on Linux. Xenomai provides sub-millisecond scheduling accuracy even for user space real-time tasks which does not exceed 100 μs in this platform. Further information on the Xenomai framework can be found in [34].

Additionally, to ensure that there is no software-wise mismatch between the nodes of the system, all of the nodes operate on the same disk image which is prepared using squashFS. During boot, this image is loaded into the memory and is mounted as the root directory using unionFS. Running the entire system on a filesystem located in RAM minimizes disk access latencies which promotes the timeliness of the system. Furthermore, since this image is re-loaded every time the system boots and changes are never written back to the original disk image located on the CF card, it's guaranteed that all nodes are always operating on the same set of software regardless of any failures that can during run-time.

5.2 W-MBPNCS Implementation

W-MBPNCS consists of sensor, controller and actuator nodes implemented as Linux executables made of real-time user-space Xenomai tasks. These nodes communicate with each other on dedicated ports using UDP over the 802.11 ad-hoc network. In order to reduce the number of computers required, sensor and actuator nodes of the system are co-located in the same computer which hosts the DAC and quadrature counter boards. However, since the sensor and the actuator never directly interact with each other, this beneficial simplification does not alter the behavior of the system in any way. For the experiments with ambient wireless traffic, a traffic generator which is identical to the nodes of the W-MBPNCS is placed strategically in the middle of two other nodes. An overview of the system is given in Fig. 5.5.

TCP vs. UDP

It's a well known fact that Transmission Control Protocol (TCP) introduces a considerable amount of overhead for establishing and maintaining a socket connection

and controlling the transmission of packages which in turn expresses itself as latency in transmission of data packets [35]. On the other hand, User Datagram Protocol (UDP) operates on simple data packages called datagrams that are used to communicate data back and forth and spends no extra time to establish, maintain and control a connection. Thus, in order to achieve better latency figures this thesis uses UDP as the transport protocol over IP.

Ad-Hoc vs. Infrastructure

Using IEEE 802.11, nodes can either associate to a central access point and communicate with each other through it or form a spontaneous network among themselves using ad-hoc mode without the need for a access point. In infrastructure mode all unicast data packets typically follow a two-hop route: source to access point, access point to destination. This essentially doubles medium access latencies and introduces additional latencies at the access point. Since W-MBPNCS requires the minimum possible latency and does not require any additional facilities provided by an access point (e.g. authentication, secure communication, etc.), ad-hoc mode of communication is used in this thesis.

5.2.1 Sensor

The main functionality of the sensor node is implemented in the periodic *sensorTask* which is executed at 100 Hz. After initializing the quadrature counter, *sensorTask* periodically calculates the position of the plant using quadrature counter (Figs. 5.6, 5.7) and sends this information to the controller node in a *sensorPacket* (Fig. 5.8). *sensorTask* also sends all relevant data through a pipe to *sensorMonitor* which saves this information in a binary log file for post-processing.

5.2.2 Controller

Controller is implemented with two periodic (*controllerTask*, *refTask*) and one aperiodic task (*listenTask*). *refTask*'s sole responsibility is to change the reference input for the ongoing experiment at an arbitrary period. Meanwhile *listenTask* receives incoming *sensorPackets*, performs initialization of the *controllerTask*, lowers the

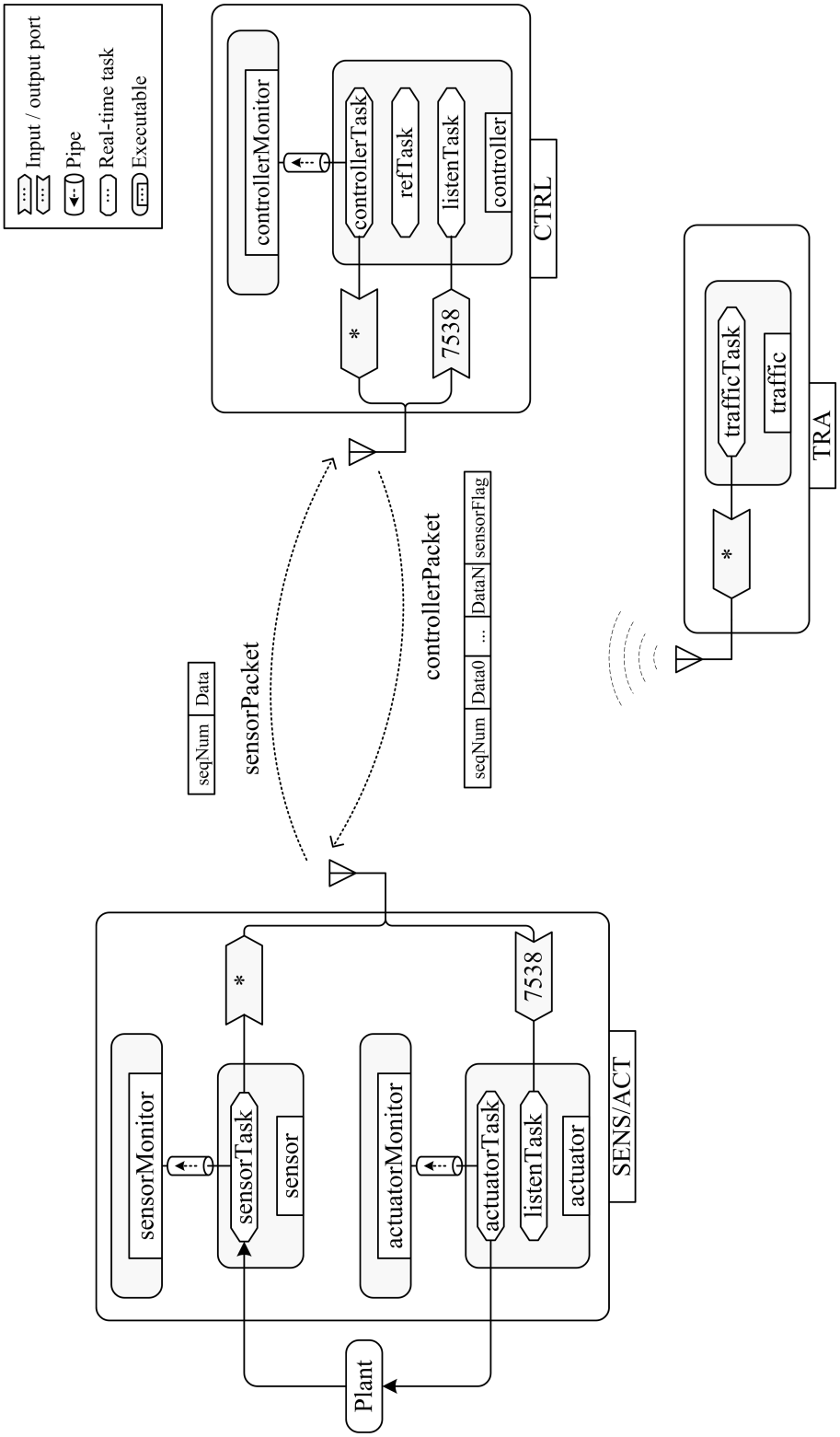

Figure 5.5: W-MBPNCS implementation

Address	0x216	0x214	0x213	0x212	0x211	0x210
Access	W	R/W	R	R	R	R
Description	Command Register	Enable / Busy Check	Counter Byte 3 (MSB)	Counter Byte 2	Counter Byte 1	Counter Byte 0 (LSB)

Figure 5.6: Relevant quadrature counter registers (Base address = 0x210)

Command Byte	0x84	0x88	0x90	0x20	0x00
Description	Read counter 0	Clear counter 0	Clear all counters	LED on	LED off

Figure 5.7: Relevant quadrature counter commands

Bytes	0-7	8-15
Format	Double	Double
Description	Sequence Number	Plant Output (rads)

Figure 5.8: Structure of *sensorPacket*

Bytes	0-7	8-15	16-415	416-423
Format	Double	Double	Double	Double
Description	Sequence Number	Control Signal	Control Signal Predictions	Sensor Flag

Figure 5.9: Structure of *controllerPacket*

sensorFlag if a packet is lost or raises it if a packet meets its deadline and saves the newly received plant output to be used by the *controllerTask*. When executed, *controllerTask* first calculates the control signal for the current time step, updates the plant model and saves the state of the model for the next sampling time to be used in case the *sensorPacket* is lost. Next, *controllerTask* runs the controller algorithm using the state of the plant model as the plant state estimate, saves this control signal estimate and applies this to the plant model. *controllerTask* repeats the last routine 50 times resulting 50 control signal predictions into the future. Finally, *controllerTask* prepares a *controllerPacket* (Fig. 5.9) using the control signal, control signal predictions and *sensorFlag* and sends this packet to the actuator.

5.2.3 Actuator

Actuator implementation consists of one periodic (*actuatorTask*) and one aperiodic task (*listenTask*). Similar to the case in controller, actuator's *listenTask* is also responsible for initializing the system properly and dropping delayed packets whereas *actuatorTask* implements the actuator functionality given in Chapter 3 and updates the control signal applied to the plant through the DAC board as given in Fig. 5.4.

5.2.4 Packet Loss Model Implementations

In order to be able to evaluate their effects in a controlled and reproducible way, both Gilbert/Elliot and uniform packet loss models are implemented in the nodes of the W-MBPNCS. Before transmission of each data packet, each node runs its own channel model and either transmits the packet or drops it deliberately according to the state of the model. The execution overhead of the loss models is negligible for both the sensor node ($< 0.5\%$ execution time of the sensor task) and the controller node ($< 2.5\%$ execution time of the controller task).

5.3 COMAC Implementation

COMAC protocol is implemented as a *hierarchical state machine* (HSM), meaning that it follows a finite state machine (FSM) approach with *state* nesting. Hierarchical state nesting allows new states to be defined in terms of their differences from a

parent state enabling action and transition reuse through inheritance. A common problem in conventional FSMs is the *state explosion* problem where several actions and transitions are common to all states but nevertheless have to be redefined at each state causing the complexity of the FSM grow much faster than the complexity of the system it describes. HSM approach not only solves the problem of *state explosion* in complex designs with multi-mode operation such as COMAC (transmit, relay, receive), it also results in more robust and intelligible implementations that are easier to verify and debug. In the following, implementation of COMAC protocol is discussed in greater detail.

5.3.1 Hierarchical State Machine Implementation

Following an HSM approach is beneficial only on the long run since the initial overhead is considerable: the need for a flexible implementation framework and an HSM engine. To meet this end a flexible framework and a robust HSM engine are designed which build on the *signals*, *actions*, *states*, *state handlers* and *transitions* abstraction. *Signals* are stimuli for states. When a *signal* is *posted* to the COMAC HSM, *HSM engine* delivers this signal to the *state handler* of the current *state*. At this point, the *state handler* may respond to the *signal* with an *action*, a *transition* or by passing the signal to its parent state. A *signal* that is not handled by any other *state handler* is discarded by the top-most parent state after being passed along to it by the nested states. If a *signal* triggers a transition, *HSM engine* properly sends the *special signals* to all states on the *transition* path:

- Entry: The signal posted to a state handler when entering to a state.
- Exit: The signal posted to a state handler when exiting from a state.
- Init: The signal which triggers the initial transition of a state.
- Probe: The signal which is used to determine the parent of a state and raises neither actions nor transitions.

As the *entry* and *exit signals* can be used to ensure some behavior by design, the *init signal* allows more intelligible and natural implementations. To prevent any unintentional mismatches between the design and the implemented HSM, the *HSM*

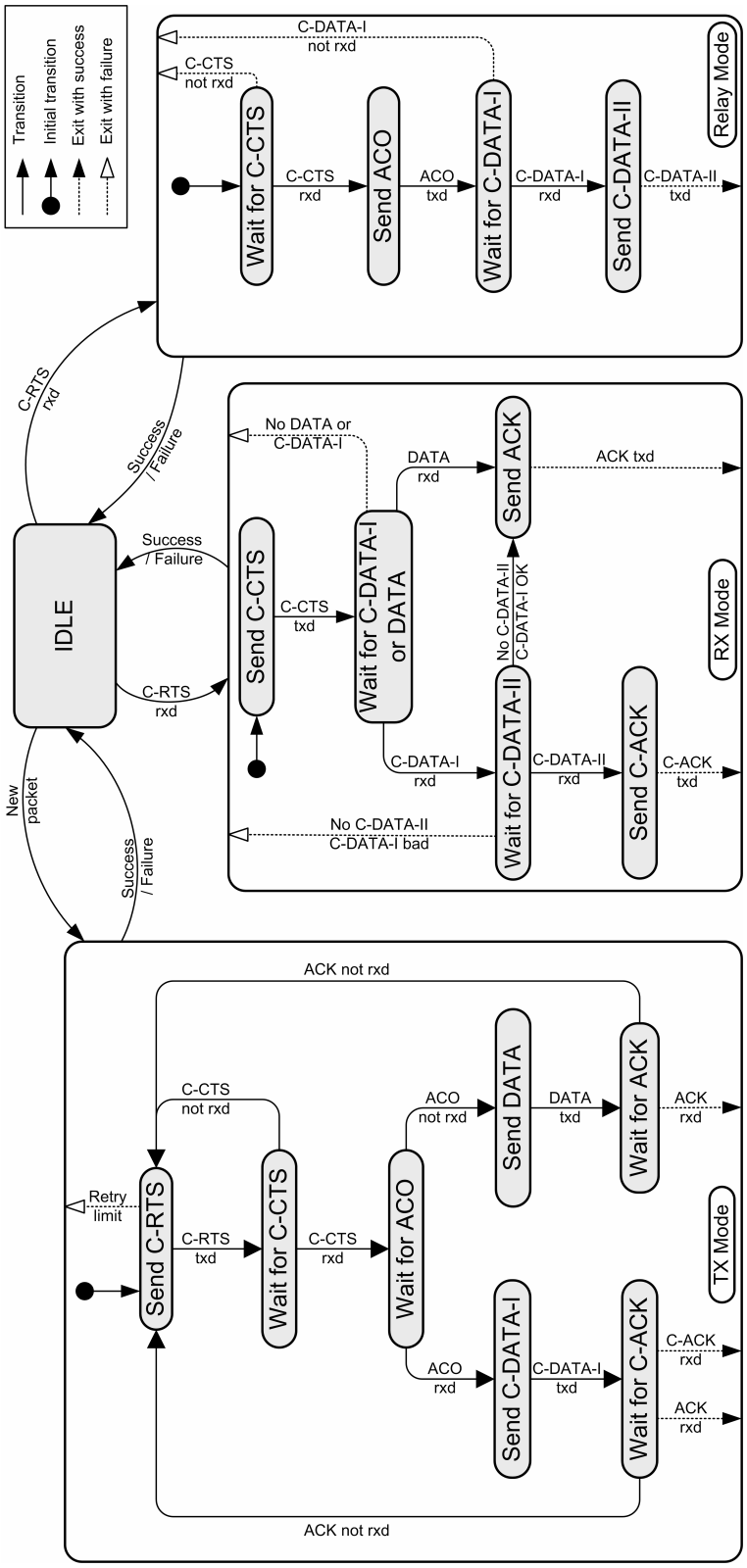

Figure 5.10: State diagram of COMAC implementation.

engine runs by the following rules and raises exceptions or warnings if any one of them is violated:

- During a transition all states on the path are exited until the parent state of the target state is reached, and all states on the path to target state are entered.
- Transition from a nested state to another nested state under the same parent does not exit the parent.
- Transition from a nested state to its parent exits itself, its parent and re-enters the parent.
- Transition to self exits the state and re-enters it.
- *Init* signal is sent only to the target of a transition. Non-empty *Init* handlers overridden during a transition raise an exception.
- Simultaneous pending signal and transition raise an exception as behavior in such a case is undefined.
- State transition and signal posting in the same signal handler is illegal (above).
- Transitions are not allowed during handling of *Entry* and *Exit* signals.
- Signal posting is not allowed during handling of the *Init* signal.

COMAC implementation encompasses a state machine structure with various fields to make the realization of the HSM and the protocol itself possible. Most relevant fields of this structure is given in Tb. 5.1 where fields related to debugging and hardware are omitted for the sake of brevity. In Lst. 5.1 a short state handler is given to illustrate how concepts of *signals*, *state nesting* and *state handling* are put into action.

Listing 5.1: TX mode send_DATA state handler

```
comacState comac_tx_send_DATA(comacSignal sig) {

    // Signal handlers
    switch(sig) {
        case entry:
            // Reset NAV
```

```
                COMAC_RST_NAV();
                // Send DATA frame
                comac_send(COMAC_DATA_QCL);
                // Debug message
                COMAC_INFO("DATA txd.\n");
                // Post signal
                SIGNAL(DATA_txd);
                return 0;

            case init:
                // No initial transition
                return 0;

            case exit:
                // No exit actions
                return 0;

            case DATA_txd:
                // DATA txd, proceed to
                // wait_for_ACK state
                TRANSITION(tx_wait_for_ACK);
                return 0;

            default:
                // TX state is the parent state
                return tx;
        }

}
```

5.3.2 Emulation of Rayleigh Fading and MRC

Since floating point operations and file access is unavailable in kernel space, exponential random variables of the Rayleigh fading model are stored as a hard-coded look-up table (LUT) of fixed point values inside the kernel module. This LUT consists of exponentially distributed random values with a mean of 10^{-3} which corresponds to the case where transmission power is 1 mW and distance is 1 m. Before these values are imported to the kernel module, they are scaled by 10^{16} to provide adequate precision during fixed point operations. When a frame is received, the receiving node first checks the MAC address of the sender to find the distance between. The distance between each of the three nodes are controlled by the three module parameters: D_1_2, D_1_3 and D_2_3 as given in Fig. 5.11, whereas the effect of distance is controlled by the D_exp parameter which corresponds to the path loss exponent in the channel model. Next, a LUT access is performed if time elapsed since last LUT

Table 5.1: Relevant fields of the COMAC_hsm struct

Type	Name	Description
HSM Related Fields		
comacSignal	signal	Last signal posted to HSM
comacSignal	pendingSignal	Pending signal to be posted to HSM
comacState	state	Current state of the HSM
comacState	pendingState	Next state of the HSM in case of a pending transition
stateHandler	handler	Pointer to state handler function
MAC Address Fields		
unsigned char [6]	addrTx	Transmitting STA of the current COMAC exchange
unsigned char [6]	addrRx	Receiving STA of the current COMAC exchange
unsigned char [6]	addrRelay	Relay STA of the current COMAC exchange
unsigned char [6]	bssid	BSSID of the local (this) STA
unsigned char [6]	addrMine	Self address of the local (this) STA
Transmit Queue Fields		
int	queueTx_cnt	Transmit queue buffer count
struct sk_buff*	queueTx_head	Transmit queue head pointer
struct sk_buff*	queueTx_tail	Transmit queue tail pointer
Duplicate Detection and Recovery Fields		
unsigned char	txSeqNum	Sequence number of the ongoing exchange in the S node
unsigned char	txRetryFlag	Flag to indicate if this transmission is a retry in the S node
struct seqnum_t[3]	seqNumCache	Sequence number cache that holds the sequence number of each STA
unsigned char	rxDupFlag	Flag to indicate if the received frame is a duplicate to be discarded
Module Parameters		
unsigned int	Mac_mode	MAC mode (0:COMAC, 1:RTS/CTS, 2:DATA/ACK)
unsigned int	Ch_model	Channel model enable/disable switch
unsigned long long int	P_tx	Transmit power in mW
unsigned long long int	D_exp	Distance exponent
unsigned long long int	D_1_2	Distance between nodes 1 and 2
unsigned long long int	D_2_3	Distance between nodes 2 and 3
unsigned long long int	D_1_3	Distance between nodes 1 and 3
unsigned long long int	Rx_thr_data	SNR threshold for received data frames
unsigned long long int	Rx_thr_ctrl	SNR threshold for received control frames

access has exceeded channel coherence time (T_c) which is chosen to be 8 ms so that the channel remains correlated within a sampling period of W-MBPNCS. This T_c value also corresponds to the reasonable scenario where the source or the dominant obstacles move at a speed of $\sim 2.5\ m/s$ with respect to the receiver. Later, the raw SNR value from the LUT is scaled with P_t, D_exp and the corresponding distance parameter to obtain the SNR of the incoming frame. Whether a frame is dropped or not is determined after a comparison with the relevant threshold parameter where Rx_thr_data parameter corresponds to receiving SNR threshold for DATA packets which are transmitted at 54 Mbps and RX_thr_ctrl corresponds to receiving SNR threshold for CTRL packets which are transmitted at 6 Mbps as given in [36]. For frames other than C-DATA-I and C-DATA-II received at D, frame's SNR value is compared with the corresponding threshold parameter and the frame is dropped if its SNR is smaller than the threshold value.

For C-DATA-I and C-DATA-II frames received at D, MRC implementation is responsible to determine whether the payload can be decoded successfully or not. At the final stage of COMAC protocol , C-DATA-I and C-DATA-II frames are transmitted simultaneously by S and R and are decoded using MRC at D. However, as this is not possible to achieve using commercial-off-the-shelf hardware, in the implemented COMAC protocol C-DATA-II frame is transmitted only by R as given in Fig. 5.12 whereas S remains silent during this period and MRC is emulated in software. When a C-DATA-I or C-DATA-II frame is received at D, SNR calculation is performed as usual, but instead of dropping or accepting the frame directly, obtained SNR values are saved in the COMAC HSM structure. After receiving the C-DATA-II frame, D checks to see if the sum of the SNR values of C-DATA-I and C-DATA-II frames is greater than or equal to the receiving SNR threshold for data frames. If so, transmission is deemed to be successful and D responds with a C-ACK frame. Otherwise, D remains silent and the other nodes operate accordingly.

5.3.3 Flow of Execution

COMAC HSM resides in the kernel modules of the wireless NIC driver and runs in both kernel space and interrupt space. Running inside a kernel module has both advantages and disadvantages: a code in kernel space and interrupt space generally

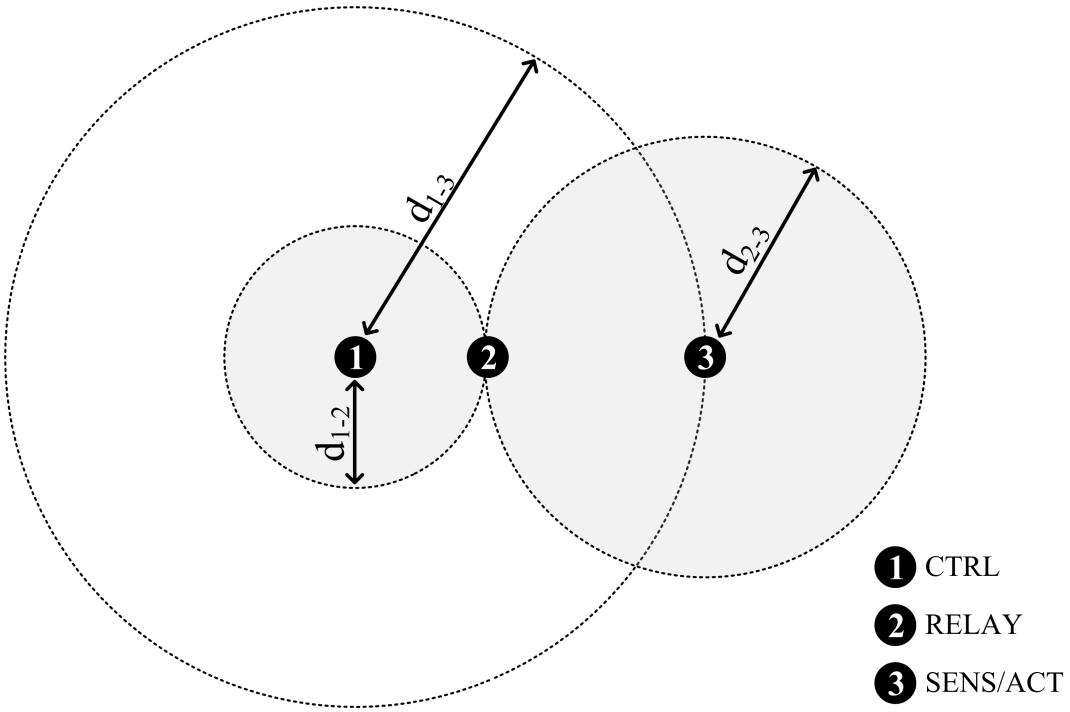

Figure 5.11: Placement of wireless nodes

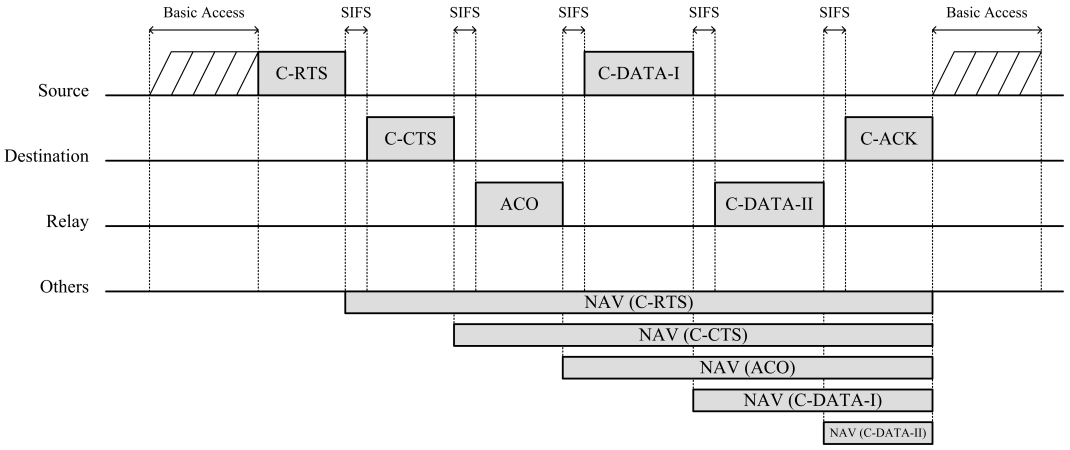

Figure 5.12: Frame exchange of COMAC implementation

enjoys lower latencies since it has a higher priority than user-space tasks but it is also much harder to code with higher chances of kernel panics and suffers from many limitations such as lack of various system calls (filesystem access, etc.) and no support for floating point operations. Nevertheless, implementing the COMAC protocol right between the driver and the protocol stacks of the operating system ensures that introduced additional latency is kept at a minimum which overweighs any other disadvantages. Details related to inner workings of the Linux kernel is out of the scope of this thesis and the interested reader is referred to [35, 37–41].

As the hardware initializes, COMAC HSM initialization routine is called which saves the MAC and BSSID addresses of the local host, allocates the COMAC frames and initializes various other fields of the HSM structure. The main execution context of COMAC HSM is the receive interrupt service routine of the rt2x00 driver. The function responsible for processing incoming frames is modified such that it first checks to see if the incoming frame is a valid COMAC frame and runs the HSM engine if it is. All other unicast DATA frames are dropped whereas management frames and broadcast DATA frames are passed to the protocol stack (mac80211) to ensure proper operation of the network (e.g. ad-hoc network formation, beacon generation, address resolution through ARP, etc.). If a frame needs to be transmitted (e.g. C-CTS in response to C-RTS) when the HSM engine is running it is written to the transmit queue and the relevant state handler waits for a reply for the timeout interval. If no frame is received in the timeout interval COMAC HSM takes the proper action depending on its state. On the other hand, an incoming frame causes the state handler to return only to be called again for the newly received frame. At the end of a successful COMAC transaction the received frame is passed to the protocol stack as a regular IEEE 802.11 frame.

In the transmit path, the function responsible for writing frames to wireless NIC transmit queue is modified such that all frames to be transmitted is written to COMAC HSM's transmit queue instead and HSM engine is called. The state machine checks the queue to see if there are any waiting frames and returns if it finds none. Any management or broadcast DATA frames waiting on the queue are transmitted using regular IEEE 802.11 bypassing COMAC to ensure proper operation of the network, whereas unicast DATA frames cause COMAC HSM to

Bit:	25	24	23	22	21	20	19	18	17
Frames to filter:	ACK CTS	Broadcast	Multicast	Version Error	to DS	Not to me	Control	Physical Error	CRC Error

Figure 5.13: TXRX_CSR0 register frame filter control bits

enter transmit mode and initiate a COMAC exchange. When a transmission is complete the post-transmission ISR is called which is responsible for deallocating the space used by the transmitted frames. This ISR is modified such that it does not deallocate COMAC frames which makes it possible to reuse COMAC frames over and over again saving the time that would be lost for allocating for each COMAC transaction.

To be able to compare the performance of COMAC protocol with regular IEEE 802.11 frame exchanges using the same framework and the same channel model, several branches are added to the implemented HSM which are controlled by the Mac_mode parameter. This parameter controls the state flow of the HSM to realize a total of three modes of operation: COMAC, IEEE 802.11 RTS/CTS and IEEE 802.11 DATA/ACK.

5.3.4 Low Level Issues

Most soft-MAC implementations suffer from big compromises due to lack of knowledge about the hardware in use. In order to implement COMAC protocol faithfully five key points have to be addressed: reception of overheard frames, modification of NAV and medium access behavior, suspension of ARQ (ACK messages), timeouts during receive and implementation of duplicate detection and recovery in software.

Reception of Overheard Frames

Most wireless NIC hardware filters out various frames such as frames not addressed to themselves and control frames, but also provide the functionality to configure these filters. Using the TXRX_CSR0 (Fig. 5.13) register, these filters are disabled such that frames not addressed to a STA is passed to the driver so that COMAC STAs can receive frames not addressed to them as required by the protocol.

Bits	Access	Description
0:14	R/W	New value in μs.
15	WO	Update flag.
16:31	RO	Current value in μs.

Figure 5.14: NAV reset through MAC_CSR15 (addr:0x303c, width:32)

Modification of NAV and Medium Access Behavior

After the C-RTS and C-CTS exchange between S and D, R must send an ACO frame to S and later a C-DATA-II frame to D. However, this is impossible in the IEEE 802.11 standard due to virtual carrier sensing mechanism: R must set its NAV vector to the duration value of the C-RTS and wait for this duration. This behavior is altered by manipulating the NAVs of the wireless NICs through their MAC_CSR15 registers (Fig. 5.14). A macro that resets the NAVs of the STAs is called whenever a STA needs to sends a frame although it was not the recipient of the previous frame.

Once a COMAC exchange is initiated with a C-RTS, other frames of the exchange must be transmitted one SIFS time after one another. However, IEEE 802.11 standard mandates that all STAs trying to access the medium for the first time must use the basic access mechanism inducing additional medium access latencies that conflict with the COMAC protocol. This behavior is brought closer to the ideal case by modifying MAC parameters of each following COMAC frame such that DIFS time is set to 0 and CW_{min} and CW_{max} are set to 0 and 2 respectively minimizing medium access latencies within an ongoing COMAC exchange.

Suspending ARQ

COMAC protocol has a frame exchange format reminiscent of the RTS/CTS mechanism and its own acknowledgement procedure, thus IEEE 802.11 acknowledgement mechanism must be disabled at both the receiving and the transmitting STAs to prevent latencies caused by these ACK exchanges and superfluous repetitions of individual frames of an exchange. In COMAC implementation, the ARQ mechanism is disabled through a special flag in transmit descriptor part of the socket buffer at the transmitting STA and through the ACK policy bits of the QoS control field at

the receiving STA. Failing to activate these two modifications simultaneously either causes the transmitting STA to hang waiting for the ACK or causes the receiving STA to send extra ACK frames disrupting the frame exchange.

Receive Timeouts

During a COMAC exchange a frame can be lost due to collisions or channel quality. The receiving STA which is waiting for a frame has to have a way to tell whether a frame has been received within an interval or not. Thus, a timeout mechanism must be designed so that the transmitting STA quits waiting for a reply after some time and attempts a retry. To meet this end, the interrupt request vectors of the wireless NICs and descriptor fields of the receive queues which are filled by the wireless NIC upon reception of a frame are polled for a pre-determined time. If the expected frame is not received within this interval for some reason, the protocol can go on with regular execution of the state machine instead of being stuck at one particular state.

Duplicate Detection and Recovery in Software

Duplicate detection and recovery mechanism of IEEE 802.11 is based on the sequence number field and the retry flag bit of the MAC header. In IEEE 802.11, each STA has a counter to generate sequence numbers which is incremented only when a frame is acknowledged by the receiving STA. If the ACK frame is not received within an SIFS time, the frame is retransmitted with the same sequence number and its retry flag asserted. This scheme may cause duplicates of a single frame to be sent to the receiving STA. To filter the duplicates at the receiving STA, sequence numbers and retry flags of each incoming frame are checked and any frame with an asserted retry flag and a previously received sequence number is dropped.

Wireless NICs provide the necessary functionality to detect and ignore duplicates of received frames through their sequence numbers and retry flags. However, this functionality works only for standard IEEE 802.11 exchange formats and useless in our case. Since the ACK mechanism is disabled, each transmitted frame has a different sequence number and a low retry flag which inhibits the duplicate detection and recovery functionality at the receiving STA. Even if ARQ is not disabled, this

Bits	15	14-8	7	6-5	4	3-0
Sub-field Name	Queue Size		Reserved	Ack policy		TID
Description	Retry Flag	Sequence Number of the current frame	0 : Default 1 : COMAC	01 : Disable ACK at receiving STA	Set to 1	COMAC Frame Type 0000 : C-RTS 0001 : C-CTS 0010 : ACO 0011 : C-DATA-I 0100 : C-DATA-II 0101 : C-ACK 0110 : DATA 0111 : ACK

Figure 5.15: QoS control field of COMAC frames

mechanism works only for individual frames of a COMAC exchange and not for the whole transaction. To fix this issue, a duplicate detection and recovery mechanism which uses the *queue size* bits of the QoS control field (Fig. 5.15) is implemented in software.

5.3.5 Frame Formats

Initial trials showed that using reserved protocol version values in the frame control field for differentiating COMAC frames causes receiving STAs to drop frames due to protocol version mismatch. A possible work-around is to turn off all receive filters to tell the hardware to pass all received frames to the driver. However this approach introduces a lot of overhead on the operating system since all irrelevant frames such as control frames, corrupted frames and frames of other wireless networks have to be filtered out in software. On the other hand, using a frame type of QoS DATA is mandatory to disable the acknowledgement mechanism at the receiving COMAC STA. Thus, COMAC implementation uses QoS DATA frames and utilizes the QoS control field of in the MAC header to differentiate COMAC frames from regular 802.11 frames and to implement soft duplicate detection and recovery as given in Fig. 5.15.

5.4 W-MBPNCS over COMAC

5.4.1 Utilization of Testbed Nodes

W-MBPNCS is realized using two computer boards instead of three, whereas COMAC requires at least three nodes for a successful cooperative exchange. Since experiments with ambient wireless traffic are not repeated when COMAC is used, the third node called TRA in W-MBPNCS experiments is utilized as the relay node and referred to as RELAY in experiments where W-MBPNCS nodes communicate using COMAC. Thus, nodes 1, 2 and 3 in the Rayleigh fading model of COMAC corresponds to CTRL, RELAY and SENS/ACT respectively as given in Fig. 5.11.

Since the sensor and the actuator nodes of W-MBPNCS are realized on the same physical computer and use the same wireless NIC, a mandatory assumption has to be made in the experiments where W-MBPNCS nodes communicate over COMAC: link qualities of the sensor and the actuator nodes are correlated. Nevertheless, this assumption is very reasonable as in most of the applications sensor and actuator nodes are indeed located very close to each other. Additionally, this assumption also simplifies the scenarios considered in such experiments leading to results that are much easier to interpret.

5.4.2 Latency Considerations

The implemented COMAC protocol runs in the kernel space and is not preemptible. Although Linux kernel is the lowest priority task of the Xenomai scheduler, it can block real-time tasks of the system if it remains in a non-preemptible section for too long. This problem was observed during the initial attempts to run W-MBPNCS over COMAC. The problem manifested itself through missed periods in W-MBPNCS tasks which broke the time-stamp mechanism and rendered the system unusable. To fix this issue some parts of the code were re-written for speed and the transmission retry limit which was initially set to 10 was decreased to 2.

Chapter 6

Experimental Results

6.1 Experiment Setup

During experiments an automated test method based on host/client pairs is used in order to eliminate human error. These host/client pairs communicate using TCP/IP over ethernet both to prevent any interference to the wireless communication of the nodes and for reliable execution of the tests. All nodes of the testbed run a host program called *remoteHost* which executes any incoming commands over a particular TCP port and replies with the results of the commands. TRA/RELAY node also runs a client program called *remoteClient* which is responsible for controlling all of the nodes in the testbed by sending the necessary commands to prepare the nodes for the experiments, initiate the experiments and collect the results in an organized fashion when the experiments end. Nodes of the testbed are placed 30 cm apart and a laptop is placed 1,5 m away for observing the wireless traffic, storage and post-processing of experiment data as given in Fig. 6.1.

6.2 W-MBPNCS over IEEE 802.11

In this section, results of experiments during which W-MBPNCS nodes communicate using individually addressed DATA/ACK frames are presented. These results were obtained from 4 tests each consisting of 40 experiments. Each test was repeated 10 times resulting in a total of 1600 experiments each 30 seconds long.

In the experiments conducted, mainly the performance of the W-MBPNCS is compared with the performance of a basic WNCS (b-WNCS) that behaves in the

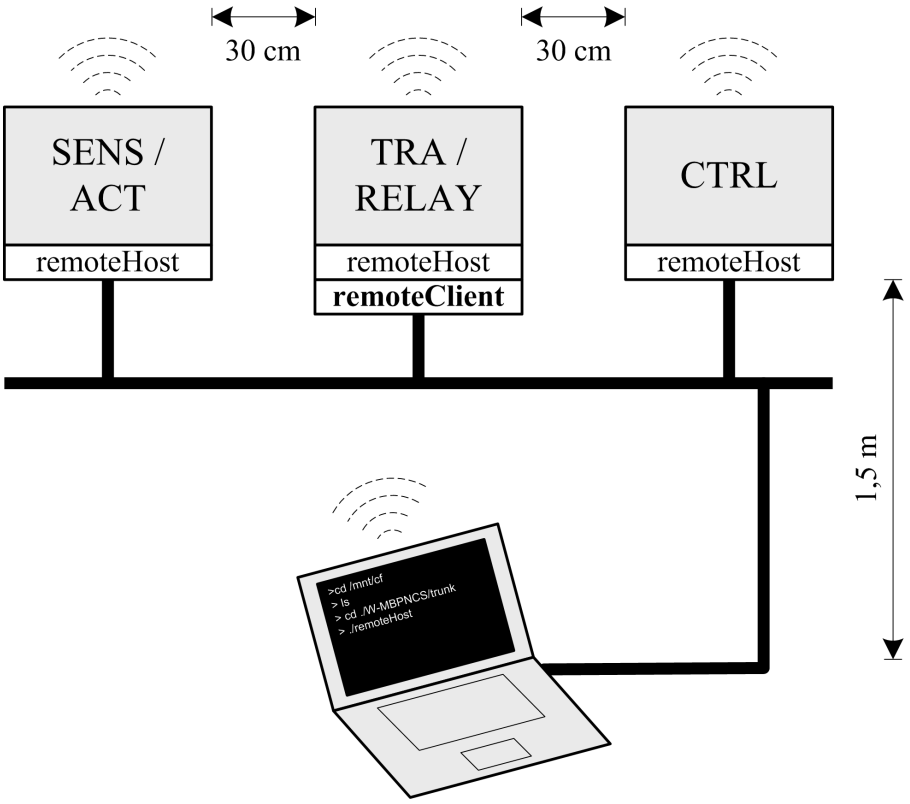

Figure 6.1: Experiment setup

following way in case of packet loss: if a sensor packet is dropped on its way to the controller, the controller does not produce any control signals and if the actuator does not receive any packets from the controller, it does not alter the last input applied to the plant. The nodes of the b-WNCS have the same protection against late coming packets (time stamps) as the nodes of the W-MBPNCS do.

Another focus of the experiments is the effect of packet loss on the performances of the compared WNCS's. Experiments both with Gilbert/Elliot and uniform packet loss models are conducted to compare the performances of the WNCS's in both cases.

As a means to observe the effect of ambient wireless traffic on the performances of the WNCS's and the improvement provided by modified MAC parameters tests both with and without constant bit-rate (CBR) traffic (750 UDP packets/s with 50 bytes of payload and duration of $648\mu s$) using both stock and modified MAC parameters (Tb. 6.1) are conducted. Size of the traffic packets' payload is chosen such that their transmission duration is less than the $1ms$ relative packet deadline between the nodes of the system. This way, the main reason of packet loss in case of contention with the traffic generator will be the back-off mechanism of 802.11b

which can be handled with MAC modifications and not the duration of the traffic packet. MAC parameters of the traffic generator are left at their stock settings in order to observe the effects of the modified MAC parameters on the performances of the nodes of WNCS's and no packet loss model is employed in the traffic generator for maximum interference. Along with the Gilbert/Elliot and uniform packet loss models, ambient wireless traffic is the 3^{rd} packet loss mechanism employed in the experiments.

Table 6.1: Modified 802.11b Parameters

Profile	DIFS	CW_{min}	CW_{max}
Stock	50	31	1023
Modified	30	0	3

The actual effect of the MAC modification is given in Fig. 6.2 which is the delay distributions of 3000 packets sent by the nodes of the W-MBPNCS under different conditions. Under ambient wireless traffic, the delay variance of the packets transmitted by the nodes with modified MAC parameters is shrunk to a great extent when compared with the delay variance of the packets transmitted by the nodes with unmodified MAC parameters. With a relative packet deadline of 1 ms, almost half of the packets miss their deadlines and are discarded under ambient wireless traffic when WNCS's operate with stock MAC parameters whereas modified MAC parameters provide an almost 100% improvement.

$$Percentage\ eRMS = \sqrt{\frac{\sum_{k=1}^{n}(\theta[k] - r[k])^2}{\sum_{k=1}^{n} r[k]^2}} \tag{6.1}$$

In the experiments, control performance of a WNCS is determined by its percentage root mean square of error (eRMS) which is a measure of the error in plant position with respect to reference position. For an experiment, percentage eRMS of a WNCS is calculated as given in Eq. 6.1 where $r[k]$ and $\theta[k]$ are reference and plant positions at time step k. In Figs. 6.3 to 6.6 performances of both WNCS's are compared with respect to their percentage eRMS averages taken over 10 identical test-runs each consisting of 160 experiments and results of the tests are presented as percentage eRMS versus employed loss model's mean packet loss rate ($\overline{PLR}_m$)

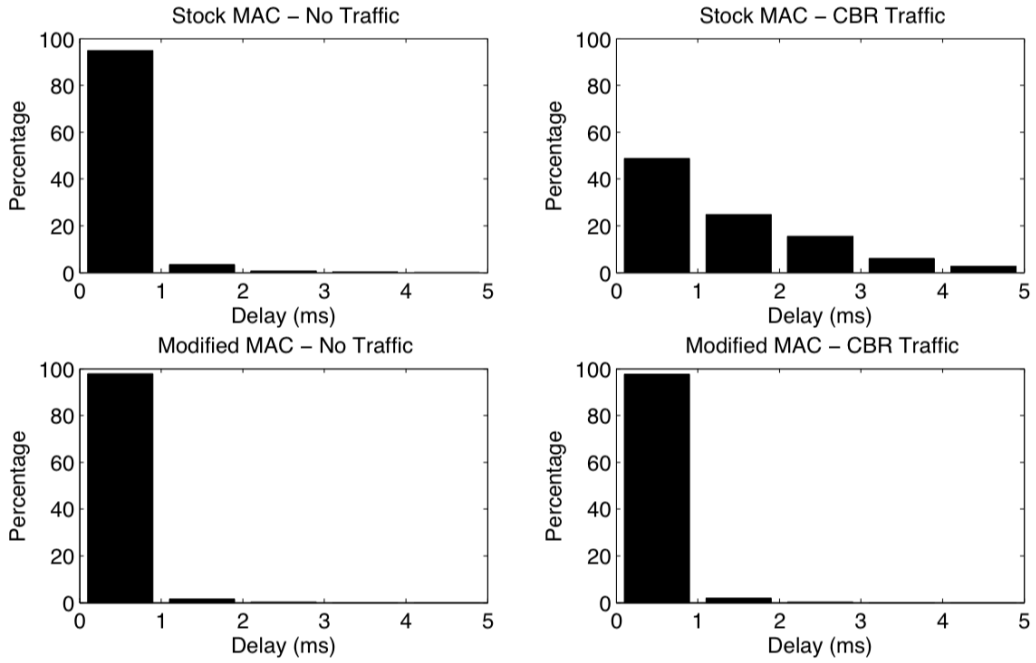

Figure 6.2: Delay distribution of W-MBPNCS packets under different conditions

plots. For the Gilbert/Elliot model $\overline{PLR}_m$ is the weighted average of P^g_{loss} and P^b_{loss} with respect to steady state probabilities of the model being in a given state and for the uniform packet loss model $\overline{PLR}_m$ equals P_{loss}. In experiments with ambient wireless traffic, additional packet loss induced by collisions is not included in $\overline{PLR}_m$ and results of such experiments are presented as additional lines in the figures. The same reference signal is used in all experiments which is a 0.5 Hz step reference with an amplitude of *2 radians* except for the additional experiment where a sawtooth reference with a slope of *4 radians/s* is used.

In the first test (Fig. 6.3), the performance of WNCS's are evaluated using standard MAC parameters under bursts of packet loss created by the Gilbert/Elliot model both with and without ambient wireless traffic. Good state packet loss probability P^g_{loss} of the model is swept from 0% to 45% at 5% increments to imitate worse than ideal channel characteristics in the good state. For the case with no traffic, the percentage $eRMS$ of W-MBPNCS is 54 at 7% $\overline{PLR}_m$ and remains under 75 up to 39% $\overline{PLR}_m$ whereas b-WNCS becomes immediately unstable under bursty packet loss. Since the percentage eRMS of b-WNCS exceeds 800 even at 7% $\overline{PLR}_m$, its curve is not included in the plot. Under ambient wireless traffic performance of W-MBPNCS degrades by at least 15%, nevertheless it still remains stable and

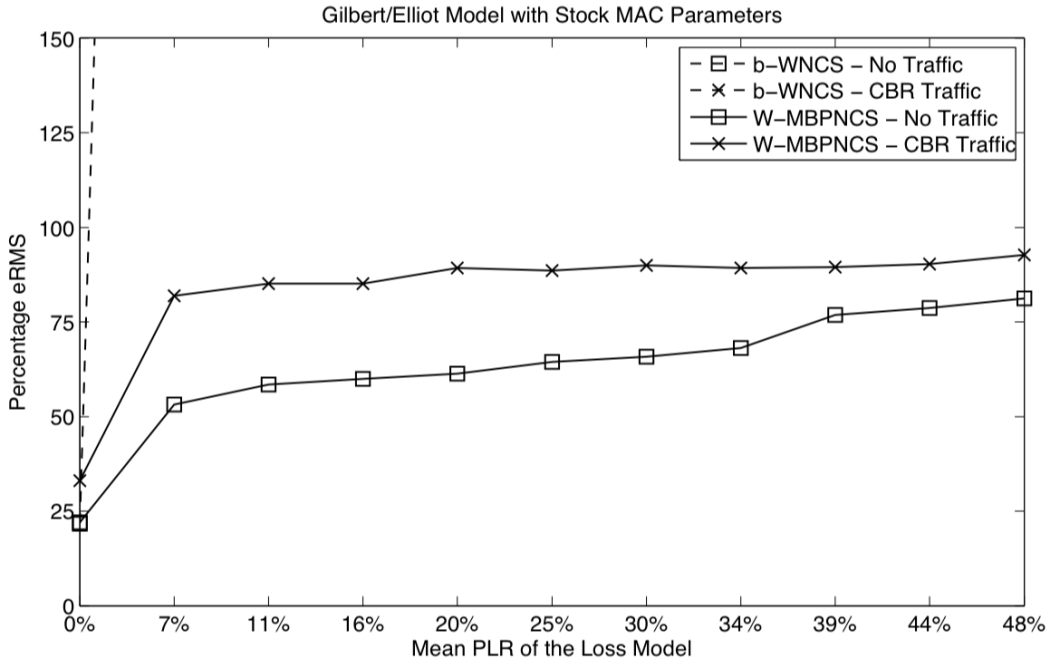

Figure 6.3: Test 1: Gilbert/Elliot model with standard MAC parameters

outperforms b-WNCS owing to its model based predictive controller. On the other hand, b-WNCS becomes unstable under ambient wireless traffic with a percentage eRMS exceeding 160 even when Gilbert/Elliot model is inactive.

In order to observe the performance gains introduced by the modified MAC parameters in the second test (Fig. 6.4), the scenario of the first test is repeated using modified MAC parameters. As expected, performances of WNCS's are not affected when there is no ambient traffic. However, under ambient wireless traffic the performance degradation of W-MBPNCS is reduced by up to 100% and b-WNCS becomes functional again when Gilbert/Elliot model is inactive owing to more deterministic channel access provided by modified MAC values. These results indicate that modified MAC parameters indeed reduce the latency that WNCS packets are subject to under ambient wireless traffic, enabling them to perform better at a given packet loss rate.

As a means for comparison between loss models, a third test (Fig. 6.5) is conducted using the uniform packet loss model by sweeping the packet loss probability (P_{loss}) between 0% and 90% with 10% increments. In the first test, W-MBPNCS suffers from relatively long periods of insensitivity to the reference due to bursts of packet loss coinciding with the transitions of the reference and b-WNCS becomes

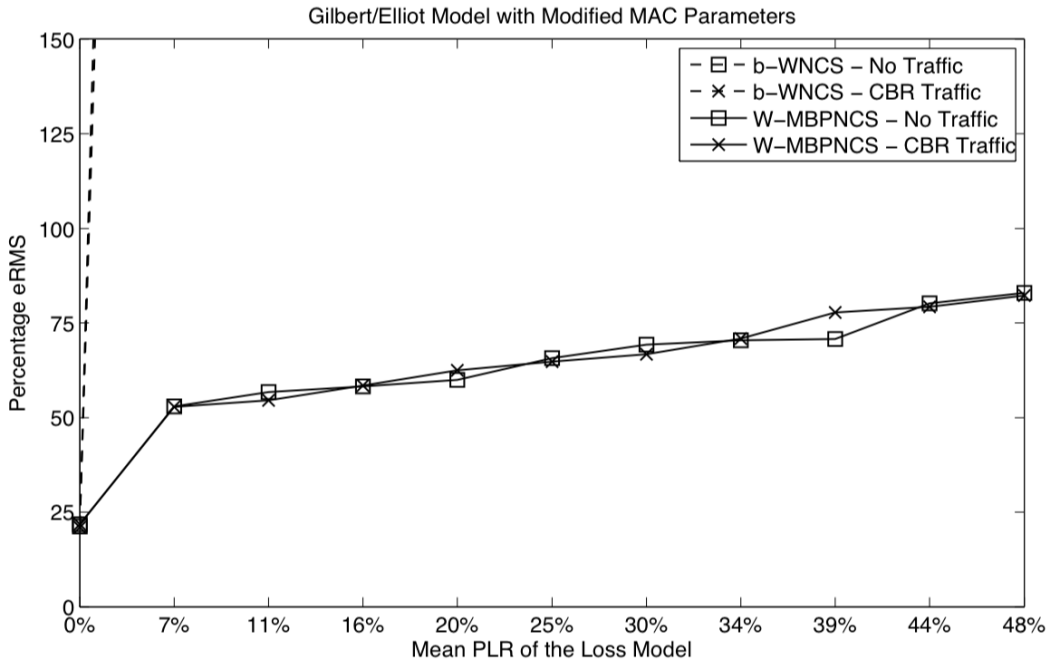

Figure 6.4: Test 2: Gilbert/Elliot model with modified MAC parameters

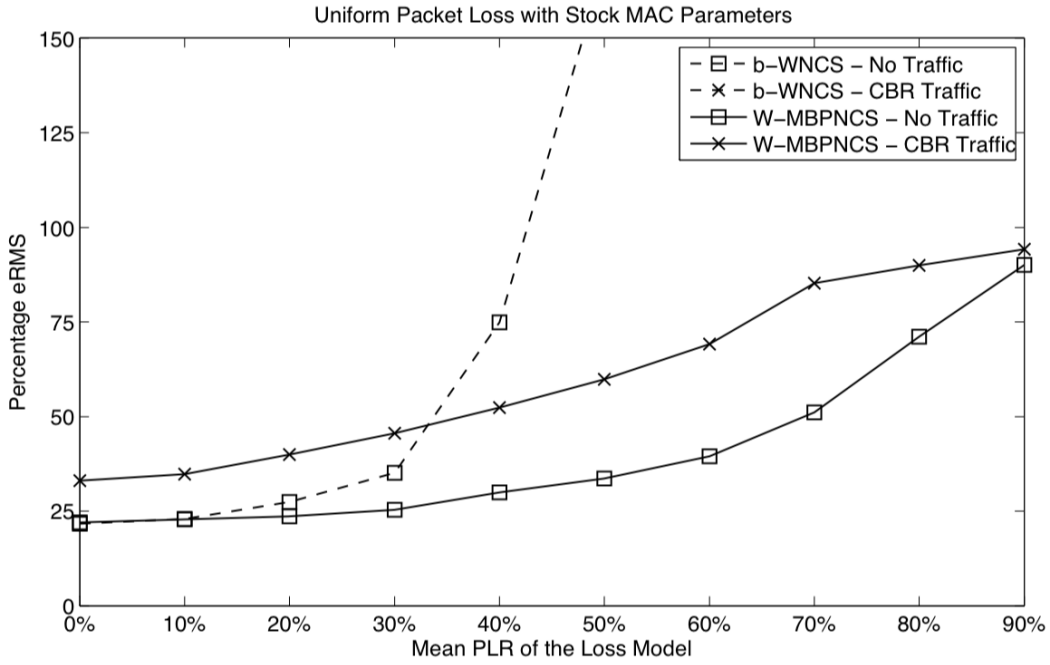

Figure 6.5: Test 3: Uniform packet loss model with stock MAC parameters

abruptly unstable under bursty packet loss. In contrast, both WNCS's perform better in this test which shows that uniform loss model results in misleading optimistic results since it does not take the correlation between packet losses into account. Nevertheless, when there is no ambient traffic W-MBPNCS's percentage eRMS remains below 50 up to 70% $\overline{PLR}_m$ whereas b-WNCS's percentage eRMS exceeds 50 around 35% $\overline{PLR}_m$. Under ambient wireless traffic percentage eRMS of W-MBPNCS remains below 50 up to 40% $\overline{PLR}_m$ while b-WNCS becomes unstable with a percentage eRMS exceeding 160 even when the uniform loss model is inactive.

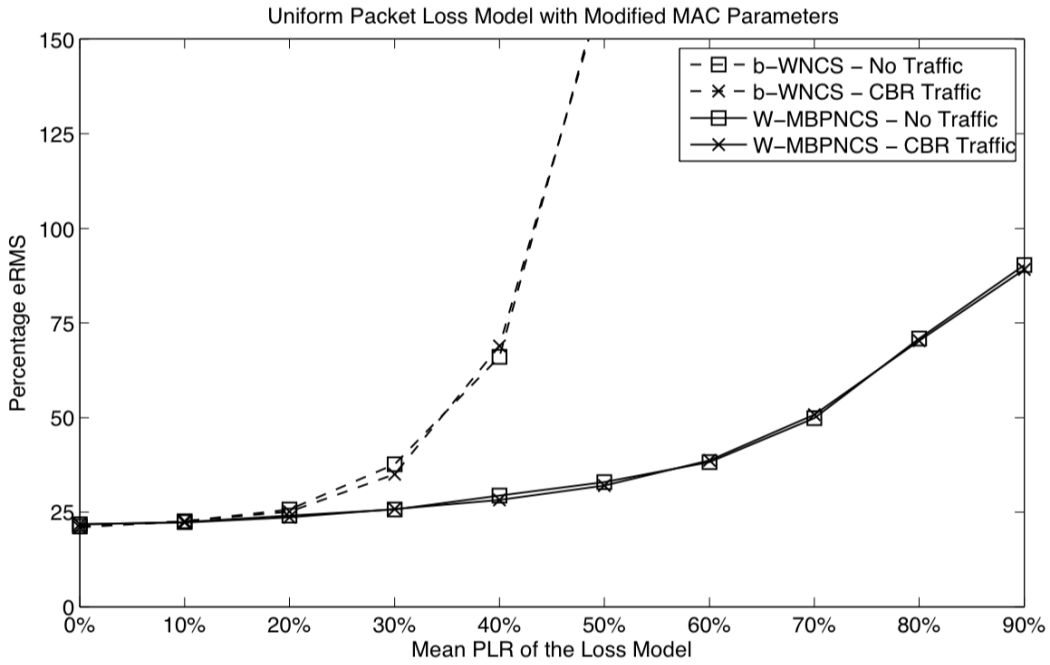

Figure 6.6: Test 4: Uniform packet loss model with modified MAC parameters

Similar to the second test, in the fourth test (Fig. 6.6) the scenario of the third test is repeated with modified MAC parameters. As expected, effect of ambient wireless traffic on W-MBPNCS's performance is reduced by up to 100% and b-WNCS becomes functional again under ambient wireless traffic.

Finally, in order to provide a better insight into the responses of both WNCS implementations to different reference inputs, two time plots of plant output (motor position) versus reference input obtained using Gilbert/Elliot loss model with a $\overline{PLR}_m$ of 7% with no ambient wireless traffic are given in Figs. 6.7,6.8. In Fig. 6.7 W-MBPNCS closely follows the reference owing to predictions produced by its controller, whereas b-WNCS's instability manifests itself as spikes in the plant output.

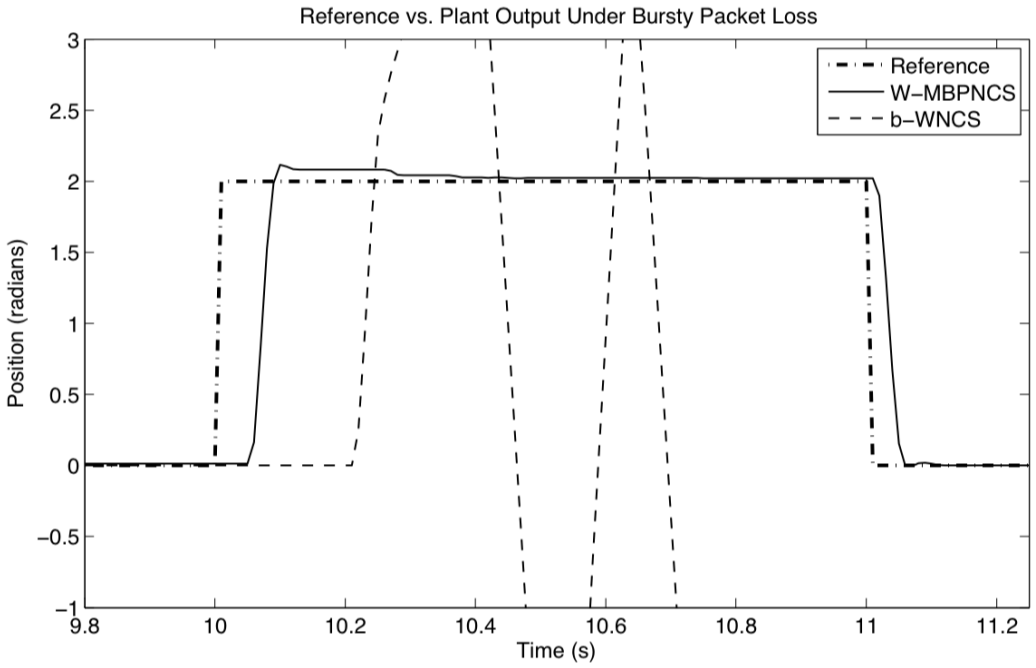

Figure 6.7: Step reference time plot

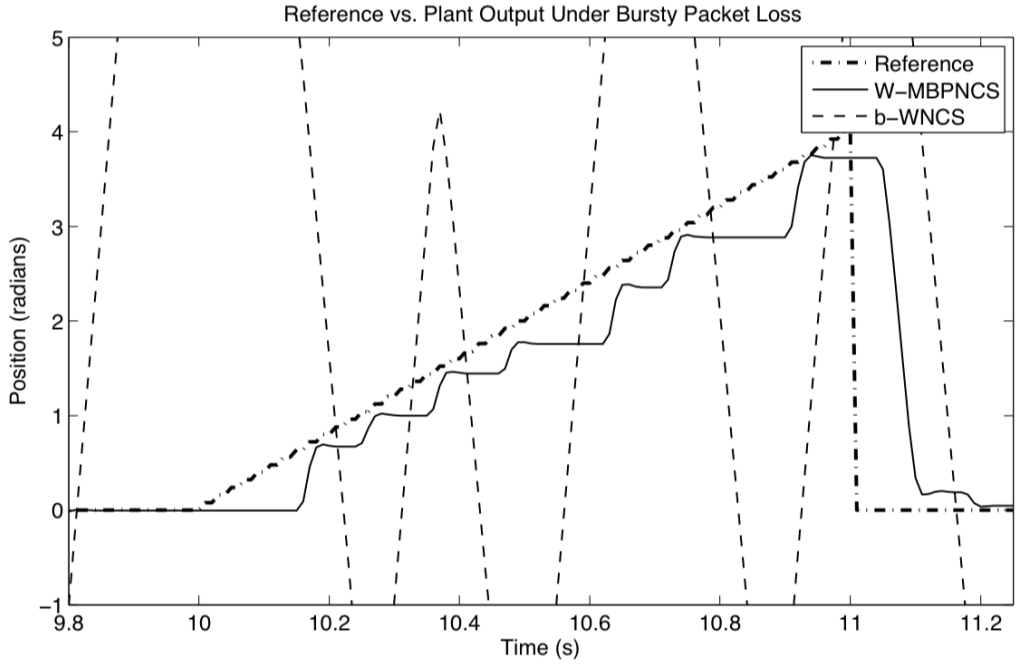

Figure 6.8: Sawtooth reference time plot

However, in Fig. 6.8, where a sawtooth reference with a slope of 4 radians/s is applied, W-MBPNCS shows some insensitivity to the tracking of the reference input due to loss of communication between the controller and actuator. Percentage eRMS values of b-WNCS, W-MBPNCS for the test presented in Fig. 6.8 are 726% and 77% respectively.

6.3 W-MBPNCS over COMAC

When W-MBPNCS nodes communicate over COMAC using the Rayleigh fading model, the most important system variable becomes the placement of the nodes. As the distance between two nodes increase, SNR at the receiver degrades increasing chances of packet errors. Thus, this section focuses on the performance of W-MBPNCS when using both COMAC and regular IEEE 802.11 frame exchange over a Rayleigh fading channel for different node distributions.

In the following, the nodes of the system are assumed to be placed along a straight line as given in Fig. 5.11 with varying distances between each other. A total of 50 scenarios made of 5 sets of 10 scenarios are considered for experiments using COMAC. Within each set the relative position of the relay with respect to other nodes is constant and the distance d between the CTRL and SENS/ACT nodes are swept from 40 m to 85 with 5 m increments. In the first set RELAY node is positioned $d/6$ away from the controller and this distance is incremented by $d/6$ for each set reaching $(5 * d)/6$ at the fifth set. For experiments with individually addressed DATA/ACK frames (regular IEEE 802.11 frame exchange), since there is no relay, a single set made of 10 scenarios is considered where distance d between the CTRL and SENS/ACT is swept from 40 m to 85 m at 5 m increments. Each experiment is repeated 10 times resulting in 600 experiments.

Fig. 6.9 illustrates how COMAC improves the packet loss rate between the nodes of W-MBPNCS. Using the regular IEEE 802.11 DATA/ACK frame exchange, more than 55% of the packets are lost at CTRL when CTRL and SENS/ACT are 60 m away and packet loss rate goes to 100% as d increases. On the other hand, when the nodes communicate over COMAC and the RELAY is in the middle of CTRL and SENS/ACT, only 7% of packets are lost at CTRL when d is 60 m and loss rate

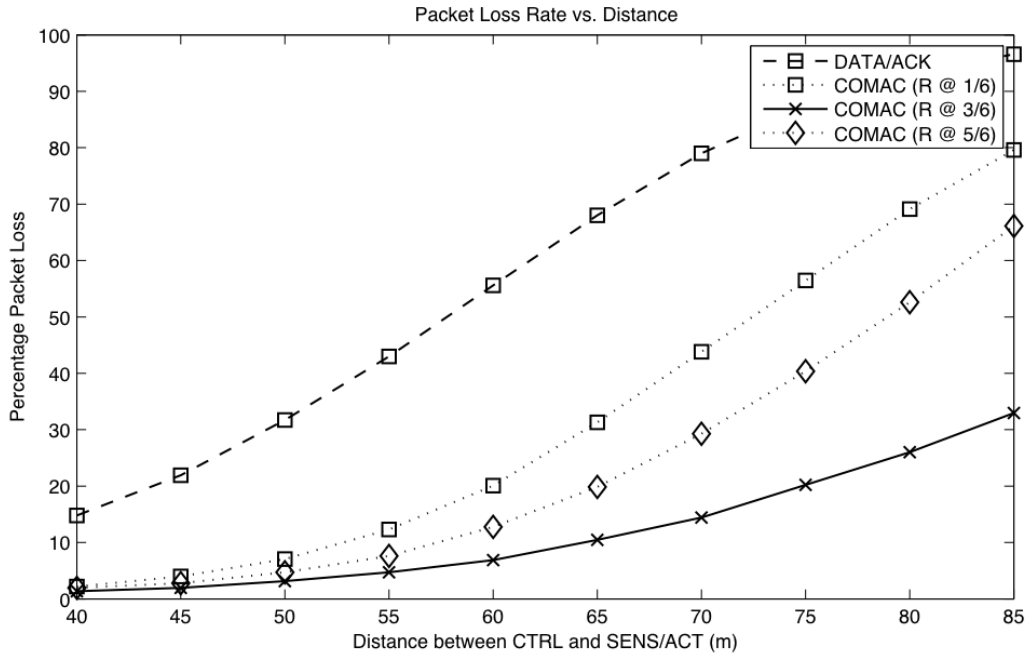

Figure 6.9: Packet loss rate at the controller node vs. distance

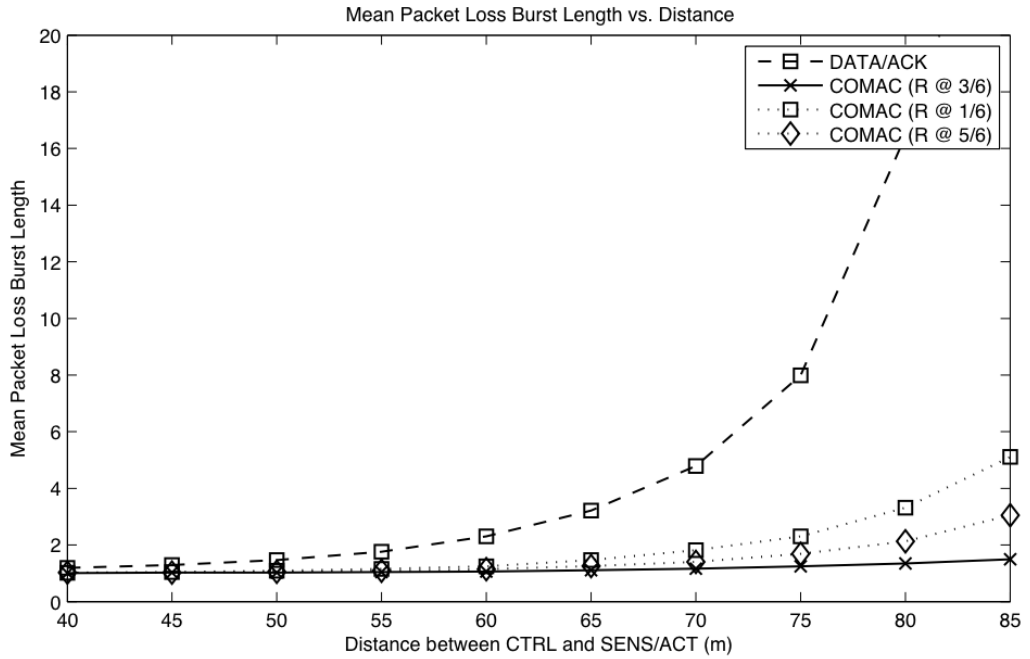

Figure 6.10: Mean packet loss burst length at the controller node vs. distance

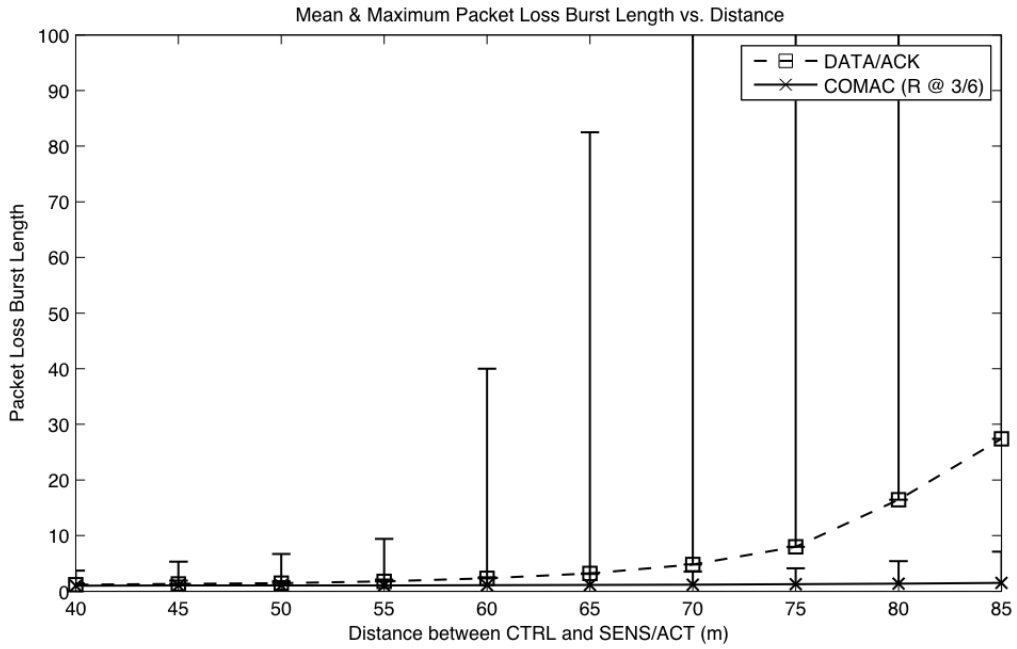

Figure 6.11: Mean and maximum packet loss burst lengths at the controller node vs. distance

is 33% when d is 85 m. Nevertheless, as W-MBPNCS is affected more by bursts of packet loss, mean and maximum packet loss burst lengths (Figs. 6.10, 6.11) are more suitable metrics for quantifying the improvement in the wireless link from the perspective of the control system. Using DATA/ACK, mean packet loss burst length at CTRL is 5 when d is 70 m and exceeds 20 when d reaches 85 m, whereas when the nodes communicate over COMAC, it always remains below 2 when the RELAY is in the middle and never exceeds 6 for other cases. More importantly, when nodes communicate using DATA/ACK maximum packet loss burst length at CTRL increases superlinearly with d and exceeds 40 when d is 60 m as given in Fig. 6.11. For a 100 Hz control system such as the one used in the experiments, this corresponds to 0.4 seconds of insensitivity to reference input which renders the system unusable for most cases. On the other hand, variance in packet loss burst length is greatly reduced when COMAC is used and maximum packet loss burst length at CTRL never exceeds 8 when RELAY is in the middle.

Fig.6.12 illustrates how the improvement in the wireless link translates to improvement in controller performance. When nodes use DATA/ACK, controller system performance degrades as d increases and percentage eRMS exceeds 70% for d

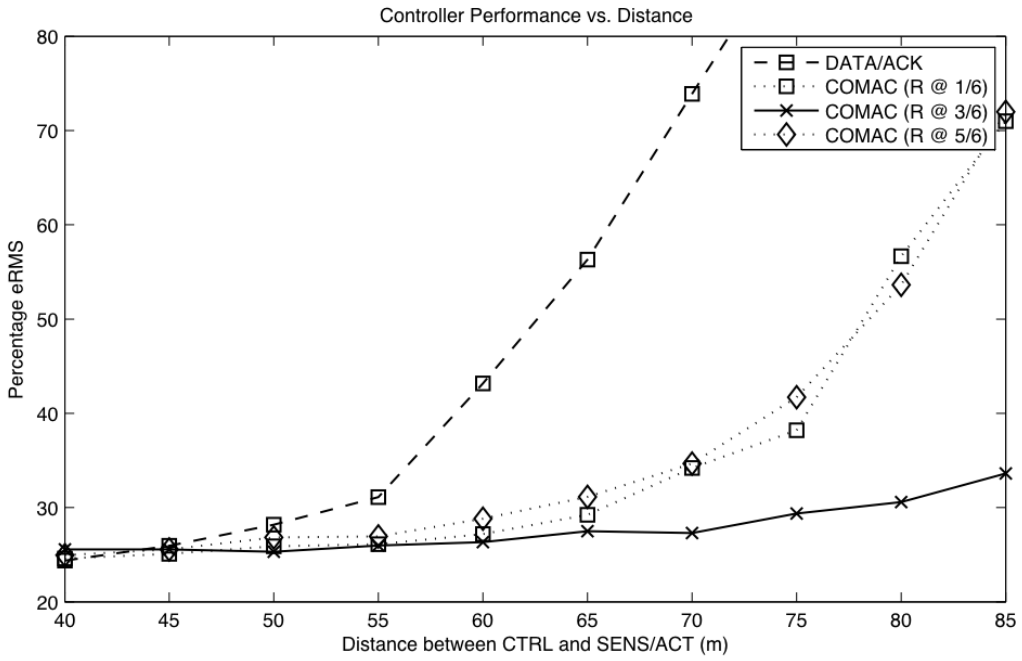

Figure 6.12: eRMS vs distance

greater than 70 m whereas when COMAC is used and the RELAY is in the middle, controller performance is almost independent of d up to 85 m as percentage eRMS remains below 35% in all scenarios.

In Figs. 6.9, 6.10 it is apparent that performance of COMAC degrades when the RELAY is not in the middle of CTRL and SENS/ACT and worst performance is achieved when the RELAY is closer to CTRL. This can be explained as follows: When R is closer to S chances of initiating a cooperation is higher but as both S and R are away from D, SNR's of C-DATA-I and C-DATA-II at D will be lower decreasing chances of successful cooperation. Thus, such a placement results in higher packet losses when compared with the case where R is in the middle. On the other hand, when R is closer to D chances of initiating a cooperation is lower since SNR's of C-RTS and ACO frames exchanged between R and S will be lower. Since performance gains achieved by the utilization of the COMAC protocol directly depend on cooperation, the worst packet loss figures are observed when R is closer to D. Nevertheless, as W-MBPNCS is a closed loop system (e.g. SENS/ACT sends a packet to CTRL in response to which receives a packet from CTRL) both cases cause a degradation in the controller performance. Fig. 6.13 illustrates the effect of the position of RELAY on the performance of the control system. For small d this

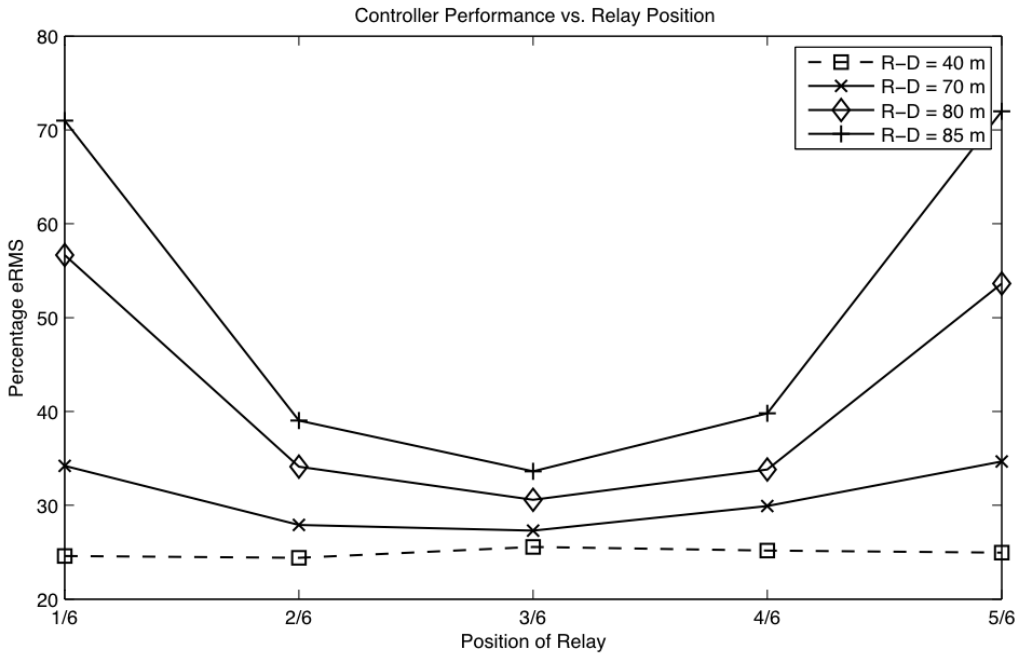

Figure 6.13: eRMS vs. relay position

effect is negligible as expected, however as d increases positioning RELAY in the middle of CTRL and SENS/ACT gives the best controller performance.

Finally, Fig. 6.14 illustrates how W-MBPNCS benefits from the improvement in the wireless link quality provided by COMAC when RELAY is in the middle, d is 70 m and reference position applied to the controller is a sawtooth signal with a slope of 4 radians/s. When W-MBPNCS nodes communicate using the IEEE 802.11 DATA/ACK mechanism the system remains insensitive to changing reference during bursts of packet loss, whereas the plant follows the reference position closely when COMAC is utilized.

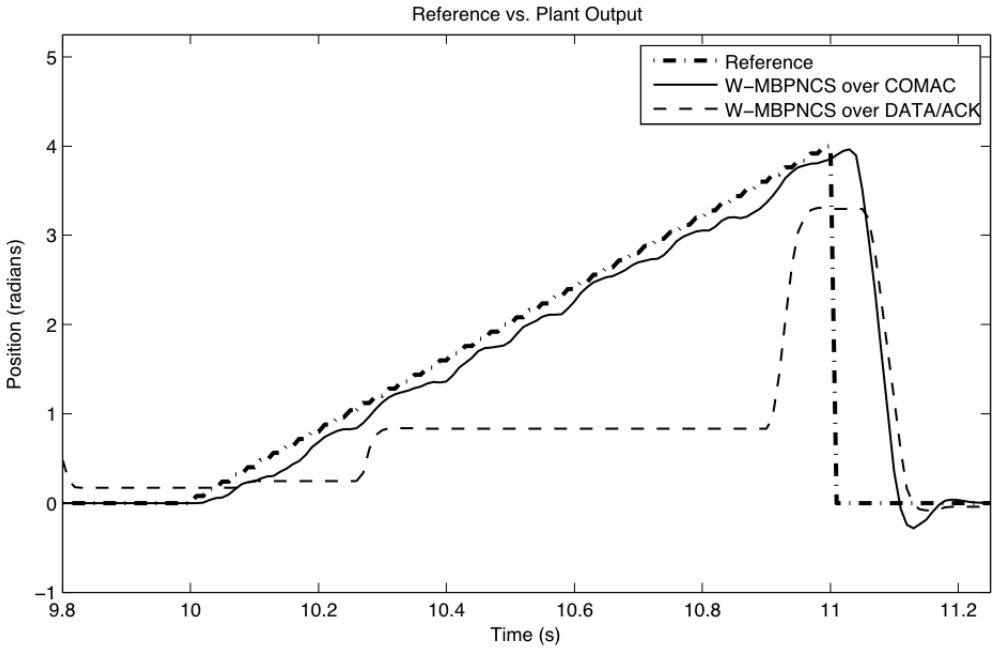

Figure 6.14: Sawtooth reference time plot

Chapter 7

Conclusion

In this work, Wireless Model Based Predictive Networked Control System (W-MBPNCS) and a faithful Cooperative Medium Access Control (COMAC) protocol implementation are presented and their performances are evaluated through extensive experiments on a test platform.

W-MBPNCS is a robust wireless networked control system (WNCS) which operates over a wireless ad-hoc network formed between its nodes. In order to minimize packet delays and losses due to collisions caused by ambient wireless traffic, W-MBPNCS takes advantage of modified medium access control parameters for higher priority medium access. Relative packet deadlines defined on each node of the system introduce an upper bound on packet latency by discarding late arriving packets, even though the network does not provide such a bound. As a means to tolerate intermittent packet losses, the controller of the W-MBPNCS employs a model of the plant to be used in prediction of future control signals which are applied to the plant by the actuator. Thus, unbounded packet latency is reduced to tolerable packet loss and the negative effect of the packet loss is minimized by the model based predictive controller.

As W-MBPNCS approaches the problem of operating an NCS over a wireless network from the control perspective, COMAC focuses on the physical and medium access control (MAC) layers of the wireless communication protocol. Through cooperation of the neighboring nodes of a wireless network, COMAC achieves higher packet success rates and longer transmission ranges in a Rayleigh fading channel when compared to non-cooperative communication. In turn, this improvement in wireless link quality leads to improvement in WNCS performance.

First, aiming position control of a DC motor, the performance of the proposed W-MBPNCS is experimentally evaluated in comparison with a basic WNCS (b-WNCS) over an IEEE 802.11 ad-hoc network. In order to produce realistic scenarios during experiments, a Gilbert/Elliot loss model is employed to imitate the bursts of packet loss in the wireless channel and a traffic generator is used to generate wireless traffic to disrupt the communication between the nodes. W-MBPNCS outperforms b-WNCS in all test cases and its percentage eRMS is shown to remain below 60% under ambient wireless traffic and bursts of packet loss with a mean model packet loss rate of 16% owing to its modified MAC parameters, imposed relative packet deadlines and model based predictive controller while b-WNCS is inoperative under such conditions.

Next, a distance-aware Rayleigh fading model is used to emulate a typical wireless channel in an industrial setting and performances of W-MBPNCS when communicating both using IEEE 802.11 and COMAC over such a channel are compared for different node placement scenarios. W-MBPNCS over COMAC outperforms W-MBPNCS over IEEE 802.11 in all test-cases and its controller performance remains virtually insensitive to the distance between the W-MBPNCS nodes up to 85 m as its percentage eRMS always remains below 35% whereas percentage eRMS of W-MBPNCS over IEEE 802.11 exceeds 70% when distance between the controller and sensor/actuator reaches 70 m.

The proposed W-MBPNCS is applicable to the industry since all components of the system are readily available whereas COMAC protocol requires some special hardware at the receiver side and a programmable or a custom wireless network interface card for an efficient implementation. Nevertheless, significant performance gains achieved by the COMAC protocol point out that cooperation is a strong alternative for improving the reliability of industrial wireless networks and WNCS's.

Bibliography

[1] IEEE 802.11 Working Group. *IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE, 2007.

[2] A. Willig, M. Kubisch, C. Hoene, and A. Wolisz. Measurements of a wireless link in an industrial environment using an IEEE 802.11-compliant physical layer. *IEEE Transactions on Industrial Electronics*, 49(6):1265–1282, December 2002.

[3] P. Wen, J. Cao, and Y. Li. Design of high-performance networked real-time control systems. *IET Control Theory and Applications*, 1(5):1329–1335, September 2007.

[4] A. Kato and K. Ohnishi. RTLinux based bilateral control system with time delay over network. In *Proceedings of the IEEE International Symposium on Industrial Electronics*, 2005.

[5] G. P. Liu, J. X. Mu, and D. Rees. Networked predictive control of systems with random communication delay. In *UKACC Intl. Conf. on Control*, September 2004.

[6] J. B. Rawlings. Tutorial overview of model predictive control. *IEEE Control Systems Magazine*, 20(3):38 – 52, June 2000.

[7] A. Onat, A. T. Naskali, and E. Parlakay. Model based predictive networked control systems. In *IFAC08 World Congress*, 2008.

[8] A. Willig. Polling-based mac protocols for improving real-time performance in a wireless profibus. *IEEE Transactions on Industrial Electronics*, 50(4):806–817, August 2003.

[9] A. Flammini, D. Marioli, E. Sisinni, and A. Taroni. Design and implementation of a wireless fieldbus for plastic machineries. *IEEE Transactions on Industrial Electronics*, 56(3):747–755, March 2009.

[10] S. Lee, K. C. Lee, M. H. Lee, and F. Harashima. Integration of mobile vehicles for automated material handling using profibus and ieee 802.11 networks. *IEEE Transactions on Industrial Electronics*, 49(3):693–701, June 2002.

[11] M. Baldi, R. Giacomelli, and G. Marchetto. Time-driven access and forwarding for industrial wireless multihop networks. *IEEE Transactions on Industrial Informatics*, 5(2):99–112, May 2009.

[12] Y. Li, C. S. Chen, Y.-Q. Song, Z. Wang, and Y. Sun. Enhancing real-time delivery in wireless sensor networks with two-hop information. *IEEE Transactions on Industrial Informatics*, 5(2):113–122, May 2009.

[13] E. C. v. d. Meulen. Three-terminal communication channels. *Advances in Applied Probability*, 3(1):120–154, 1971.

[14] J. N. Laneman, D. N. C. Tse, and G. W. Wornell. Cooperative diversity in wireless networks: Efficient protocols and outage behavior. *IEEE Transactions on Information Theory*, 50(12):3062–3079, December 2004.

[15] S. K. Jayaweera. Virtual mimo-based cooperative communication for energy-constrained wireless sensor networks. *IEEE Transactions on Wireless Communications*, 5(5):984–989, May 2006.

[16] A. Adinoyi and H. Yanikomeroglu. Cooperative relaying in multi-antenna fixed relay networks. *IEEE Transactions on Wireless Communications*, 6(2):533–544, February 2007.

[17] R. Madan, N. B. Mehta, A. F. Molisch, and J. Zhang. Energy-efficient cooperative relaying over fading channels with simple relay selection. *IEEE Transactions on Wireless Communications*, 7(8):3013–3025, August 2008.

[18] A. Sendonaris, E. Erkip, and B. Aazhang. User cooperation diversity part ii: Implementation aspects and performance analysis. *IEEE Transactions on Communications*, 51(11):1939–1948, November 2003.

[19] P. Liu, Z. Tao, Z. Lin, E. Erkip, and S. Panwar. Cooperative wireless communications : A cross-layer approach. *IEEE Wireless Communications*, pages 84–92, August 2006.

[20] J. Heo, J. Hong, and Y. Cho. Earq: Energy aware routing for real-time and reliable communication in wireless industrial sensor networks. *IEEE Transactions on Industrial Informatics*, 5(1):3–11, February 2009.

[21] T. Korakis, M. Knox, E. Erkip, and S. Panwar. Cooperative network implementation using open-source platforms. *IEEE Communications Magazine*, 2009.

[22] P. Liu, Z. Tao, S. Narayanan, T. Korakis, and S. S. Panwar. Coopmac: A cooperative mac for wireless lans. *IEEE Journal on Selected Areas in Communications*, 25(2):340–354, February 2007.

[23] M. S. Gokturk and O. Gurbuz. Cooperation in wireless sensor networks: Design and performance analysis of a mac protocol. In *IEEE International Conference on Communications*, 2008.

[24] M. S. Gokturk and O. Gurbuz. Cooperative mac protocol with distributed relay actuation. In *IEEE Wireless Communications and Networking Conference*, 2009.

[25] H. M. F. AboElFotoh, S. S. Iyengar, and K. Chakrabarty. Computing reliability and message delay for cooperative wireless distributed sensor networks subject to random failures. *IEEE Transactions on Reliability*, 54(1):145–155, March 2005.

[26] A. Ulusoy, A. Onat, and O. Gurbuz. Wireless model based predictive networked control system. In *IFAC International Conference on Fieldbusses and Networks in Industrial and Embedded Systems*, 2009.

[27] K. Pahlavan and P. Krishnamurthy. *Principles of Wireless Networks: A Unified Approach*. Prentice Hall, 2002.

[28] E. O. Elliot. Estimates of error rates for codes on burst-noise channels. *Bell Systems Technical Journal*, 42:1977–1997, September 1963.

[29] A. Goldsmith. *Wireless Communications*. Cambridge University Press, 2005.

[30] K. J. Astrom and B. Wittenmark. *Computer Controlled Systems*. Prentice Hall, 3rd edition, 1997.

[31] A. Onat, T. Naskali, E. Parlakay, and O. Mutluer. Control over imperfect networks: Model based predictive networked control systems. *IEEE Transactions on Industrial Electronics, submitted*.

[32] rt2x00 project, . URL `http://rt2x00.serialmonkey.com`.

[33] M. Franke. A quantitative comparison of realtime linux solutions. 2007.

[34] Xenomai: Real-time framework for linux, . URL `http://www.xenomai.org`.

[35] N. Matthew and R. Stones. *Beginning Linux Programming*. Wiley Publishing, Inc., 2008.

[36] Cisco aironet 1200 series access point data sheet, . URL `http://www.cisco.com/en/US/products/hw/wireless/`.

[37] S. Venkateswaran. *Essential Linux Device Drivers*. Prentice Hall, 2008.

[38] A. Rubini, J. Corbet, and G. Kroah-Hartman. *Linux Device Drivers*. O'Reilly Media, Inc, 2005.

[39] W. Mauerer. *Professional Linux Kernel Architecture*. Wiley Publishing, Inc., 2008.

[40] C. Benvenuti. *Understanding Linux Network Internals*. O'Reilly Media, Inc, 2005.

[41] D. P. Bovet and M. Cesat. *Understanding the Linux Kernel*. O'Reilly Media, Inc, 2005.