软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2017,28(5):1315−1325 [doi: 10.13328/j.cnki.jos.005144]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于共享内存的智能无人车进程间消息异步传输机制*

陈存铜[1,2], 赵君峤[1,2], 叶 晨[1,2], 邓 蓉[1,2], 管林挺[1,2,3], 李德毅[1,2,4]

[1](同济大学 电子与信息工程学院,上海 201800)

[2](同济大学 嵌入式重点实验室,上海 201800)

[3](浙江海洋大学 数理与信息学院,浙江 舟山 316022)

[4](总参第 61 研究所,北京 100081)

通讯作者: 赵君峤, E-mail: zhaojunqiao@tongji.edu.cn

摘 要: 智能无人车软件系统通常由多个功能模块组成,在模块间高效、可靠地传输传感器数据以及决策和控制信息等,是智能无人车系统运行的重要保障.目前,国内外大多数智能无人车软件系统所使用的消息传输机制均基于套接字(socket),其容易部署在分布式的控制器环境中,且能满足在较小数据量下的消息快速传输.但是,随着智能无人车集成控制器性能的提升以及环境感知手段的发展,对功能模块间传输的数据量以及带宽提出了更高的要求.现有基于套接字的消息传输机制因其受网络协议的限制,需要分块传输大数据包,不仅增加了收发双方的开销,而且还增加了消息传输延迟.提出一种基于共享内存(shared memory)的智能无人车进程间消息异步传输机制,模块间通过共享内存空间进行数据交互.共享内存空间由超级块和数据块构成,通过环形队列管理数据块收发;同时,采用原子操作提高整体性能,实现图像等大数据包的有效传输.该设计应用于智能无人车模块间通信,可以明显降低数据传输时延,提高系统吞吐量.实验结果表明,该方法针对典型大数据包(如 3MB)的平均传输时延为 2.5ms,低于 LCM 的 12ms 以及 ROS 中 Sharedmem_transport 的 3.9ms.另外,该系统的最大吞吐量达到 1.1GB/s,高于 LCM 的 180MB/s 以及 Sharedmem_transport 的 600MB/s.

关键词: 共享内存;智能无人车;进程间通信;高带宽;低延迟

中图法分类号: TP316

中文引用格式: 陈存铜,赵君峤,叶晨,邓蓉,管林挺,李德毅.基于共享内存的智能无人车进程间消息异步传输机制.软件学报, 2017,28(5):1315−1325. http://www.jos.org.cn/1000-9825/5144.htm

英文引用格式: Chen CT, Zhao JQ, Ye C, Deng R, Guan LT, Li DY. Inter-Process asynchronous communication in autonomous vehicle based on shared memory. Ruan Jian Xue Bao/Journal of Software, 2017,28(5):1315−1325 (in Chinese). http://www.jos.org.cn/1000-9825/5144.htm

Inter-Process Asynchronous Communication in Autonomous Vehicle Based on Shared Memory

CHEN Cun-Tong[1,2], ZHAO Jun-Qiao[1,2], YE Chen[1,2], DENG Rong[1,2], GUAN Lin-Ting[1,2,3], LI De-Yi[1,2,4]

[1](College of Electronics and Information Engineering, Tongji University, Shanghai 201800, China)

[2](Key Laboratory of Embedded System and Service Computing, Tongji University, Shanghai 201800, China)

[3](College of Mathematics and Information Science, Zhejiang Ocean University, Zhoushan 316022, China)

[4](The 61st Research Institute of General Staff Department of Chinese PLA, Beijing 100081, China)

* 基金项目: 中央高校基本科研业务费专项资金(20143436); 同济大学青年优秀人才培养计划(2014KJ027); 国家自然科学基金(41201379)

Foundation item: Fundamental Research Funds for the Central Universities (20143436); Program for Young Excellent Talents in Tongji University (2014KJ027); National Natural Science Foundation of China (41201379)

收稿时间: 2016-07-27; 修改时间: 2016-09-07, 2016-10-26; 采用时间: 2016-11-04; jos 在线出版时间: 2017-01-20

CNKI 网络优先出版: 2017-01-20 16:06:30, http://www.cnki.net/kcms/detail/11.2560.TP.20170120.1606.001.html

Abstract: Timely transmission of sensory data and control instructions is crucial between function modules of autonomous car systems. Socket-based message transmission mechanisms, e.g. LCM and IPC, are the de facto standard for this purpose because of their easy deployment and adaptability for distributed environment. However, they can no longer meet the increasing demands of high bandwidth for sharing large data packets, i.e. images and point clouds, among perception processing modules and the decision making module. The main reason is that socket-based mechanisms have to divide the large data packet into smaller data packets, which introduces extra costs for packing and unpacking, therefore results in high latency and low bandwidth. This paper proposes a new shared-memory-based inter-process message transmission mechanism. The shared memory segment is composed of a super block and several data blocks. The circular queues are deployed for fast scheduling data receiving and delivering. Meanwhile, the atomic operations are applied to improve the overall performance. Experimental results show that the average transmission delay of data packets of size 3MB is 2.5ms, which is drastically lower than LCM's 12ms and is also better than Sharedmem_transport's 3.9ms. The maximum throughput of this method is 1.1GB/s, which is much greater than LCM's 180MB/s and Sharedmem_transport's 600MB/s.

Key words: shared memory; autonomous vehicle; inter-process communication; high bandwidth; low latency

智能无人车软件系统通常由多个功能模块组成,如感知模块、控制模块、决策模块等[1].每个模块以进程的方式独立执行.功能模块间协调工作:通过感知道路环境,自动规划行车路径并控制车辆到达预定目标.在此过程中,模块间需要进行大量的消息交互,如:决策模块需要将控制命令发送给控制模块控制车辆运动;雷达模块需要将采集到的障碍信息传递给感知模块进行融合处理等.由于不同模块间交互的消息数据量不同,且模块间处理消息的速率差异较大,因此,为了保证无人车决策的正确性和控制的稳定性,如何在多个模块间建立高效、可靠的消息传输机制,是智能无人车必须要面对的难题,也是智能无人车的安全保障.

目前,大多数智能无人车采用基于 Socket 的消息传输机制[2]进行数据传输,如 Carmen(carnegie mellon navigation)[3],Player[4],ROS(robot operating system)[5],LCM(lightweight communications and marshalling)[6]等.其原因一方面是 Sockets 易于使用;另一方面,Sockets 容易部署在分布式的控制器环境中.Carmen 采用了 IPC(inter process communication)消息传输机制.在 IPC 中,通过 TCP 服务器和集中的 Hub 实现客户端与客户端的直接通信.Player 采用 C/S 模型实现进程间的通信,Player 服务器运行一系列的 drivers 程序,如读取感知信息、执行控制命令等.客户端通过服务器与 drivers 直接进行数据交互.ROS 早期的版本主要采用 TCP 的方式实现进程间的通信,在新版本中,增加了 UDP 和 Spread 等消息传输方式.LCM(lightweight communications and marshalling)采用 UDP 组播的方式传输数据,所有的功能模块加入同一个组播地址,订阅特定的消息.国内的虚拟交换 VirtualSwitch[7]机制同样采用了基于 UDP 组播的方式,模块间通过特定端口号实现消息传输,目前在多个智能无人车团队中得到了应用.

随着新型智能无人车环境感知技术,尤其是视觉以及多线激光传感器的广泛应用,对图像以及点云等大数据包在功能模块间的实时传输及其带宽提出了需求[8].传统基于 Socket 的消息传输机制受网络协议的限制,当传输较大数据包时需分包传输,不仅增加了收发双方的开销,而且还增加了消息传输延迟.由于智能无人车系统对实时性有很高的要求,需要在很短的时间内感知周围环境并进行决策处理,因此,需要面向智能无人车系统发展的新趋势,提出一种支持大数据包实时传输的进程间消息传输机制.

当前,高性能智能驾驶计算平台的发展使得在单个控制器中部署多个甚至全部智能无人车功能模块已成为可能[9].众所周知,在同一台主机中,共享内存是效率最高的进程间通信方式.POSIX 提供了共享内存机制实现进程间通信,但其在数据传输过程中的收发同步机制难以满足低延迟需求[10].Ach 针对降低小数据包传输的延迟、提高机器人系统的实时性的需求进行设计[11],采用锁(mutex)和条件变量(condition)实现收发双方异步传输,收发双方通过锁互斥访问同一共享内存空间,导致数据收发不能同时进行,增加了模块间的耦合.ROS 针对机器人领域大数据包的传输需求开发了基于共享内存的消息传输机制 Sharedmem_transport[12].Sharedmem_transport 调用操作系统底层函数接口创建共享内存块,通过锁和条件变量实现收发双方之间的同步.然而在整个系统中,发送方需要等待所有的接收方接受完上次发送的数据后才能发送下一个数据,虽然保证了数据的可靠传输,但可能无法及时传输新数据,导致系统的时滞;并且 Sharedmem_transport 内每个消息主题内的模块间串行传输数据,降低了系统吞吐量.

针对上述不足,本文提出一种基于环形队列的共享内存进程间消息异步传输机制,主要针对智能无人车的功能模块运行在同一控制器内,且功能模块间需要实时传输图像等大数据包的情形进行研究.针对现有方法中收发双方同步机制的不足,提出新的传输机制,旨在保证数据实时性并提高系统吞吐量.本文的主要创新点包括:支持大数据包数据(如图像)的实时传输;通过环形队列控制系统收发,降低收发双方模块间的耦合;采用原子操作,提高整体性能.

1 基于共享内存的进程间消息异步传输模型

在 Sharedmem_transport 消息传输机制中,系统申请一块共享内存空间(shared memory segment),该共享内存空间由多个共享内存块(shared memory block)组成,每个共享内存块保存特定消息主题(topic)的数据,并由 SharedMemoryBlockDescriptor 结构体表示.发送方通过消息主题在共享内存空间内找到对应的共享内存块及其结构体,将数据拷贝到该共享内存块内,并通过该共享内存块结构体内的条件变量(condition)唤醒所有订阅该消息主题的接收方,所有的接收方被唤醒后从对应的共享内存块中获取传输的数据.整个通信模型如图 1 所示.

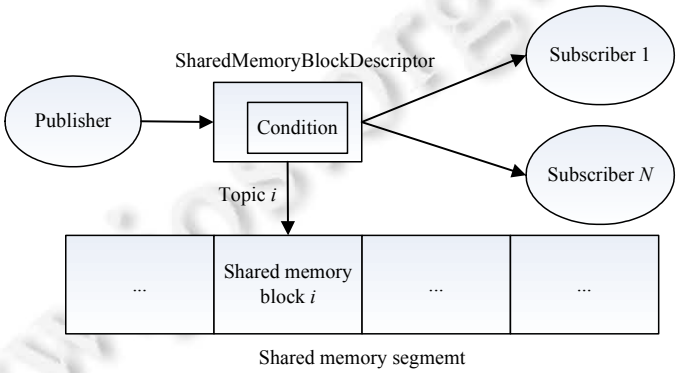


Fig.1 Communication model of sharedmem_transport

图 1 Sharedmem_transport 通信模型

Sharedmem_transport 通过锁和条件变量实现收发双方间的同步,然而在整个系统中,每个消息主题内的模块间串行传输数据,降低了系统吞吐量;并且发送方需要等待所有的接收方接受完上次发送的数据后才能发送下一个数据,增加了之后数据传输的延迟.针对上述不足,本文提出一种基于环形队列的共享内存进程间消息异步传输机制.在本系统中,每个消息主题申请一个独立的共享内存空间,为了有效管理该共享内存空间,本文借鉴了 MINIX 文件系统[13]的设计,将共享内存空间分成一个超级块和多个数据块:超级块管理数据块,数据块保存传输的数据.为了控制数据的收发,系统在超级块内为发送方创建一个写环形队列,保存空闲的数据块的索引,并为每个接收方创建一个读环形队列,保存存放数据的数据块的索引.发送方发送数据时,访问写环形队列 head 指针指向的空闲数据块,将数据拷贝到该数据块内.接收方接受数据时,遍历自己的读环形队列,获取每一个队列项指向的数据块内的数据.收发双方通过对读写环形队列的交叉操作,异步实现数据的收发.整个通信模型如图 2 所示.

该模型采用异步传输的方式,收发双方不需要建立同步机制,降低了模块间的耦合.由于传输的数据保存在数据块内,接收方每次可以接收多个数据块内的数据进行快速处理,降低了延迟,减少了丢包的可能,并且提高了系统吞吐量.需注意的是,Ach 中同样采用了环形队列[11],但该环形队列仅用来管理传输的数据,发送方将新数据覆盖旧数据,接收方从该环形队列获取最后添加的数据以保证实时性.由于收发双方利用与 Sharedmem_transport 相似的同步机制互斥访问同一环形队列,因此存在与之相同的缺陷.

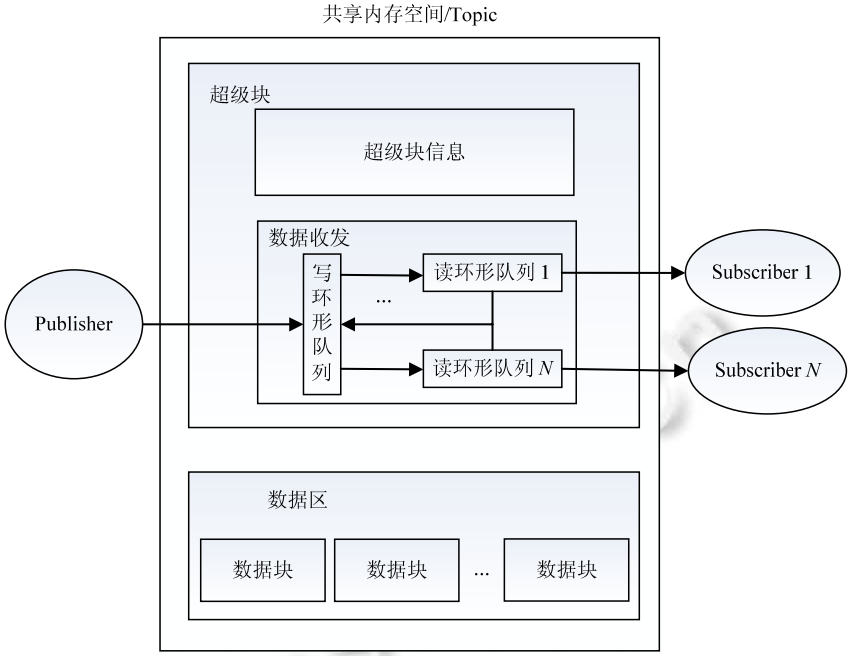


Fig.2 Communication model

图 2 通信模型

1.1 共享内存空间的组织

在本消息传输机制中,为了有效利用该共享内存空间,本文借鉴了 MINIX 文件系统的设计,特定消息主题申请的共享内存空间由一个超级块和多个数据块组成,如图 3 所示.

Fig.3 Layout of the shared memory

图 3 共享内存组织形式

超级块是该共享内存空间的核心,其由基本信息、写环形队列、读环形队列数组和数据块结构体数组组成.基本信息包含了该共享内存空间的信息,如共享内存空间的大小、数据块的大小等.在本系统中,每个消息主题只有一个发送方,因此,超级块只创建一个写环形队列,负责数据发送.由于每个消息主题可能存在多个接收方,因此超级块创建了多个读环形队列,负责对应接收方的数据接收.在共享内存空间内,每个数据块由一个结构体实例表示,所有数据块结构体的实例统一维护在数据块结构体数组内中.读写环形队列内,每个队列项保存对应数据块结构体实例在数据块结构体数组中的索引.在对环形队列操作时,通过每个队列项的索引值,在数据块结构体数组中找到对应数据块结构体实例,从而访问对应的数据块.这样一方面可以方便多个进程同时访问所有结构体实例;另一方面,当系统中存在多个接收方时,能够有效降低读写环形队列的占用空间.

每个数据块指向共享内存空间中的一段连续空间,保存该消息主题传输的数据.在本系统中,每个消息主题传输的数据结构是固定的,因此,为了提高共享内存空间的利用率,共享内存空间将超级块之外的内存空间以该消息主题传输的数据的大小为单位,分割成连续的数据块.

本消息传输机制为每个消息主题分配独立的共享内存空间,相对于 Sharedmem_transport 和 Ach 多个消息主题占用同一共享内存空间,每个消息主题可以获得更多可用的空间,从而可以更加灵活地传输数据.同时,共享内存空间借鉴了 MINIX 文件系统的设计,实现了共享内存空间的逻辑结构与物理结构的分离,提高了共享内存空间的利用率.

1.2 数据发送与接收

本消息传输机制采用环形队列管理数据块,使用固定大小的内存空间,不需要进行动态的内存释放和分配,从而降低了系统开销.在数据传输过程中,发送方每次将数据拷贝到空闲的数据块中,而接收方每次从保存数据的数据块中获取数据.为了实现有序的收发,每个读环形队列除了 head,tail 指针及队列数组外,还增加了条件变量和状态标志位.接收方等待数据时,进入阻塞态,通过条件变量将其唤醒进行数据接收.相对于 Sharedmem_transport 中所有接收方完成数据接受后通过条件变量唤醒发送方的方式,本机制中收发双方无需建立同步机制.发送方发送数据时,从写环形队列中获取一个空闲的数据块的索引,如数据块 i.根据该索引,发送方将要发送的数据拷贝到数据块 i 中,并将该索引添加到每个接收方的读环形队列的末尾.每个接收方依次遍历自己的读环形队列,根据每个队列项访问特定的数据块中的数据.当接收方获取一个数据块内的数据后,通过原子操作判断是否将该数据块的索引添加到写环形队列的末尾,以供发送方重新使用:如果是,则将该索引添加到写环形队列末尾;反之,则继续访问下一个队列项,如图 4 所示.通过对环形队列的交叉操作,屏蔽了共享内存空间的物理结构,实现了收发双方数据的异步传输,降低了模块间的耦合.

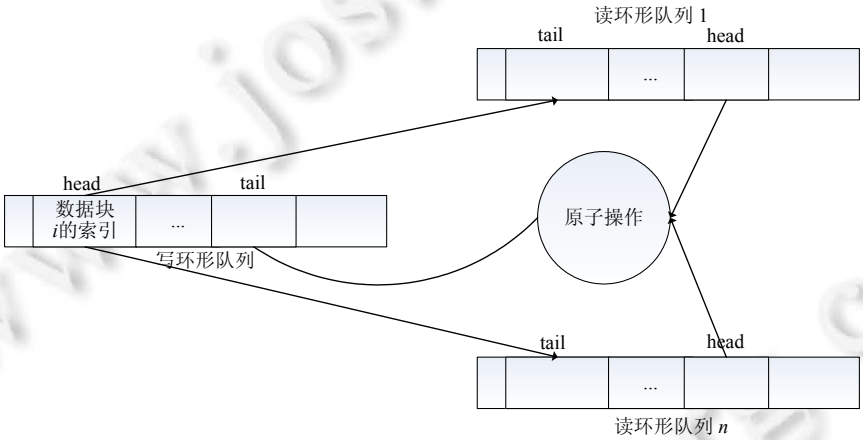


Fig.4 Model of data publishing and subscribing

图 4 数据收发模型

1.2.1 数据发送

发送方通过写环形队列发送数据.系统初始化时,写环形队列包含了所有数据块的索引.发送方发送数据时,访问超级块中的写环形队列:如果该队列不为空,则执行出队操作,初始化 head 指针指向的数据块的结构体,将要发送的数据拷贝到指定的数据块内,并将该数据块的索引添加到每个接收方的读环形队列的末尾;反之,若该队列为空,发送方丢弃此时发送的数据,从而不影响发送方的发送速率和数据的实时性,这样可以有效杜绝“过期”数据的干扰,保证决策和控制过程的稳定性.该传输策略与 LCM,Player 和 Ach 类似:LCM 不会为数据包的丢失及乱序做额外处理;Player 虽然采用 TCP 的消息传输机制,但当接收方的数据接收速率低于发送方发送速率时,Player 也允许丢包[6];Ach 接收方每次访问共享内存空间时,取其环形队列内最新的数据,忽视旧数据[11].

发送方将数据块索引添加到每个读环形队列的末尾后,通过每个读环形队列内的条件变量,唤醒对应接收方.发送方发送数据流程如图 5(a)所示.在数据发送过程中,发送方只需遍历写环形队列,不需要确认接收方是否接收到数据,降低了收发双方之间的耦合.针对部分必须保证数据绝对可靠的非实时消息传输应用,则需设计丢

包检测以及消息重传等机制[6].由于该类应用一般对传输带宽要求较低,可利用现有的传输机制,如 TCP 进行传输,因此不在本文讨论范围之内.

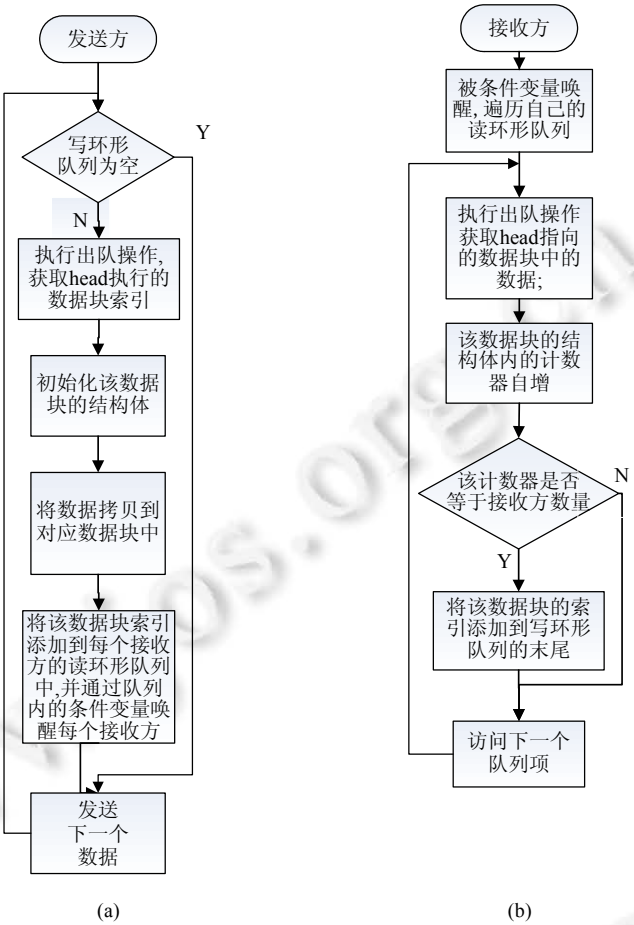


Fig.5 Routines of data publishing and subscribing

图 5 数据收发流程

1.2.2 数据接收

接收方接受特定消息主题的消息时,将该主题对应的共享内存空间映射到本进程地址空间,并从该共享内存空间内的超级块获取一个没有关联其他接受方的读环形队列.接收方被唤醒后,遍历自己的读环形队列,根据每个队列项访问指定的数据块,获取其中的数据.当一个数据块内的数据被所有接收方获取后,该数据块索引需要重新添加到写环形队列,以供发送方使用.本系统采用了引用计数的方法,当一个接收方获取一个数据块中的数据后,该数据块的结构体内的计数器就自增并判断其值是否等于所有接收方数量:如果相等,则表示所有的接收方都获取了该数据块中的数据,于是该接收方清除该数据块中的数据并将该数据块的索引添加到写环形队列的末尾;反之,该接收方执行出队操作,访问读环形队列下一个队列项.每个接收方的接收流程如图 5(b)所示.

当多个接收方同时访问同一个数据块时,可能会同时对该数据块的结构体中的计数器进行操作.为了解决多个接收方并行互斥的问题,本系统借鉴了 Lock_Free Queue[14]的设计,每个接收方通过原子操作函数 sync_fetch_and_add 等对该计数器进行操作.这时,操作系统通过锁住前端总线(FSB),阻止其他处理器对该计数器进行操作,该原子操作函数的性能是普通进程锁(mutex)的 6 倍~7 倍[15].在实际工程中,为了避免因为异常导致一个接收方不能及时释放该计数器的信号量,其他接收方进入“死等”的情况发生,接收方捕获到异常时,释放其所占

资源,保证其他接收方正常运行.而且,为了保证系统的正常运行和稳定性,系统中需添加监控模块及时重启因异常而中断的接收方.

当接受方不再需要接受该消息主题的消息时,设置对应的读环形队列内的状态标志位,发送方根据该状态标志位,不再向该接收方的读环形队列添加数据,即不再向该接收方发送数据.同时,该接受方遍历自身的读环形队列,对每个队列项指向的数据块结构体内的计数器进行自增操作,从而释放该接收方占用的数据块.

1.3 数据传输延迟及丢包分析

(1) 数据传输延迟

数据传输过程中,发送方将数据拷贝到数据块,并将该数据块的索引添加到每个接收方的读环形队列;最后,通过条件变量唤醒每个接收方开始接收数据.在此过程中,数据传输延迟受数据的内存拷贝及操作系统的进程调度的影响.当传输的数据大小增加时,发送方将数据拷贝到数据块的内存拷贝的开销随之增加,同时,接收方还需将数据块中的数据拷贝到自身进程内的缓冲区,根据统计,两次内存拷贝的开销占数据传输延迟的 60%以上,见表 1.

Table 1 Cost of memory copy (one subscriber)

表 1 内存拷贝开销(1 个接收方)

数据块大小(MB)	1	2	4	8	16
两次内存拷贝开销(ms)	0.49	1.06	1.86	2.75	7.01
消息传输延迟(ms)	0.66	1.29	2.46	4.51	10.6
内存拷贝开销占比(%)	74	82	74	61	66

此外,当发送方通过条件变量唤醒每个接收方时,每个接收方进程从阻塞态被调度执行.当接收方数量增加时,操作系统进程调度的开销随之增加.因此,综上所述,数据传输延迟随传输数据包大小和接收方数量增加而增加,如图 6 所示.

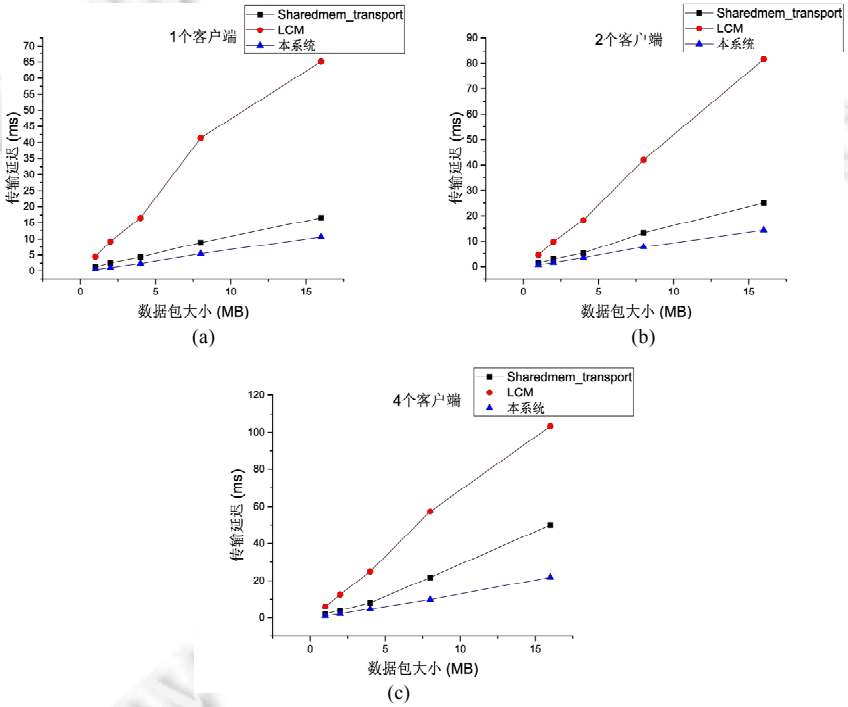


Fig.6 Latency of the message transmission

图 6 数据传输延迟

(2) 丢包

在 Ubuntu12.04 中,共享内存空间默认最大占用 32MB(该默认值可根据应用调整).因此,将共享内存空间分割为连续的数据块时,数据块数量是有限的,进而每个环形队列大小同样有限.以 N 字节的数据块和 M 字节的共享内存空间为例,若超级块占用的空间是 P 字节,则该共享内存空间最多有(M–P)/N 个数据块,每个环形队列有(M–P)/N 个队列项.

因为环形队列的大小有限,在数据发送的过程中,写环形队列可能出现没有可使用的空闲数据块索引的情况.为了保证数据的实时性,发送方会丢弃此时发送的数据,造成丢包.造成写环形队列为空的原因一方面是由于发送方的发送速率过快,导致写环形队列出队速度大于入队速度;另一方面,将数据块的索引重新添加到写环形队列的末尾,需要等待所有的接收方都获取完该数据块内的数据.在此过程中,所有的接收方需使用原子操作函数解决并行互斥问题,带来部分系统开销,从而导致写环形队列入队速率随着接收方数量的增加而下降.

在实际应用中,进行大数据传输的功能模块总数有限,因此,为了避免或有效减少丢包的产生,需要确认发送总带宽小于系统最大吞吐量.

2 实验分析

本实验模拟了智能无人车正常运行时,模块间按照固定频率传输数据的典型情形,主要测试数据传输延迟、丢包率和吞吐量.本实验的程序由 C 语言编写,运行在 Ubuntu12.04 系统中,实验机配备酷睿 i5(2.1GHz)四核处理器,2G DDR3 内存,每个共享内存空间大小是 32MB.由于 VirtualSwitch 消息传输机制暂不支持大数据包的传输,因此,本系统与国际主流的 LCM,Sharemem_transport 进行性能对比.

2.1 实验1:数据传输延迟测试

为了测试数据传输延迟,发送方以固定频率发送一定大小的数据,接收方的数量从 1 增加到 4.利用数据发送和接受的时间差计算传输的延迟.在本实验中,发送方发送 100 000 个固定大小的数据,计算平均数据传输延迟.实验结果如图 6 所示.

可见,3 种方法的数据传输延迟均随着传输数据大小及接收方数量的增加而增加,且 LCM 的数据传输延迟明显高于 ROS 及本系统.其主要原因是,LCM 采用分包传输的方式传输大数据包,从而增加了数据传输延迟.此外,本系统采用异步传输的方式,每个接收方每次可以接受多个数据进行处理,相对于 Sharedmem_transport 每次只处理 1 个数据,数据传输延迟得到进一步下降.而且随着接收方数量的上升,本方法的延迟增加比率最低.

2.2 实验2:最大吞吐量测试

在同济大学智能无人车“途灵(TiEV)”中,模块间传输的图像数据的典型分辨率大小为 1000×1000,每个像素 3 字节,每帧图像大小约为 3MB.因此,在计算系统最大吞吐量时,实验传输的数据包大小固定为 3MB,发送方总计发送 100 000 个数据包,测试在不同发送频率下的丢包率和带宽.其中,发送带宽和接受带宽的计算方式为

发送(接收)带宽=发送(接收)数据包数量×3MB/发送(接收)时间.

由于 Sharedmem_transport 必须等待所有的接收方接受完上次发送的数据后才发送下一个数据,所以在计算其最大吞吐量时,不断提高其发送频率,当其发送和接收带宽不随发送频率增加而增加时,其最大吞吐量等于此时的发送或接受带宽.在计算 LCM 的最大吞吐量时,将每个接收方的接收缓冲区设置足够大,如 4MB,使丢包率尽量低,发送方以最大的发送频率发送数据,计算此时的发送带宽以及每个接收方的接收带宽.最后,在计算本系统最大吞吐量时,不断调整发送方发送频率,计算其在不丢包时的发送和接收带宽,实验结果如图 7 所示.

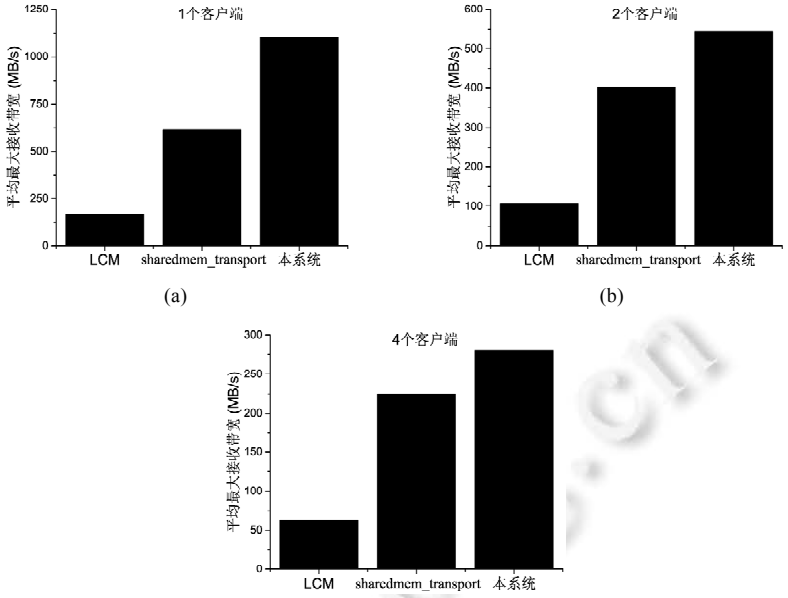


Fig.7 Experimental result of throughput

图 7 吞吐量实验结果

3 应 用

同济大学“途灵”智能无人车的视觉系统由车位检测模块、车道线检测模块、障碍物探测模块、SLAM 模块等功能模块组成,每个功能模块以进程的方式独立运行,如图 8(a)所示.

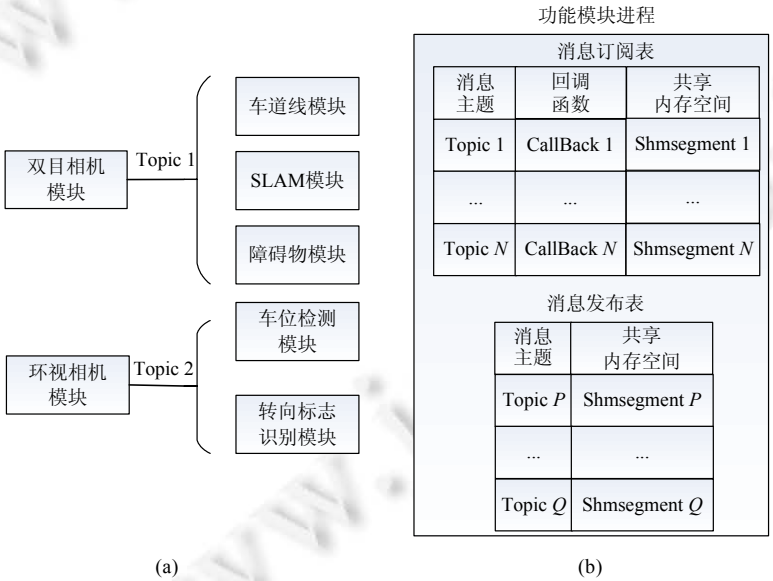


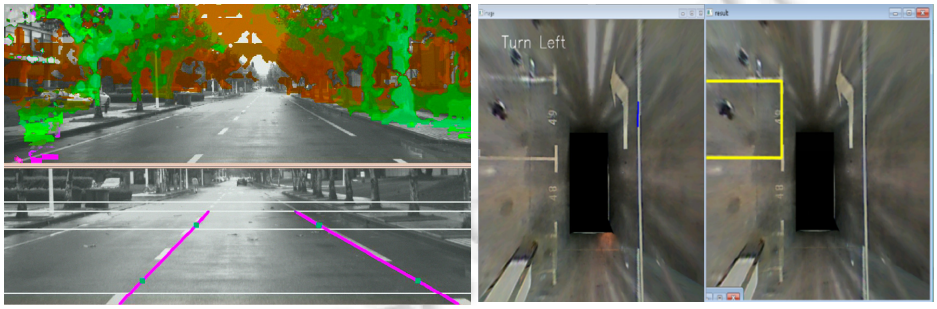
Fig.8 Application model

图 8 应用模型

“途灵”视觉系统中的每个功能模块都采用 Publish/Subscribe[16]模型进行数据的发送和接收,因此,每个功能模块进程需要包含消息订阅表和消息发布表.消息订阅表包含该进程订阅的所有消息主题、每个消息主题对应

的回调函数及每个消息主题使用的共享内存空间.消息发布表包含发布的消息主题以及对消息主题使用的共享内存空间,如图 8(b)所示.

基于本文提出的机制,相机模块能够将采集到的图像数据,发送到各功能模块中进行即时处理,部分实验截图如图 9 所示.图 9(a)显示的是双目障碍物检测模块和车道线检测模块实时运行的实验结果,其中上侧为检测到的障碍物,下侧为识别到的车道线.两个模块同时接收由双目相机模块采集并利用本机制发送的的图像数据进行处理.图 9(b)为车位检测模块和转向标志识别模块在运行过程中实时获取环视相机采集的图像信息并进行处理的实验结果,其中,左侧显示检测出的转向标志,右侧方框内区域显示检测到的车位.其中,双目相机采集图像的频率最高为 30Hz,每帧双目图像 6MB,最高带宽约为 180MB/s,4 路环视相机采集图像的最高频率为 25Hz,每帧拼接图像 3MB,最高带宽约为 75MB/s.本系统的带宽和延迟均能满足系统需求.



(a) (b)

Fig.9 Application demonstration

图 9 应用结果展示

4 总结与展望

消息的快速、可靠传递机制,是智能无人车的生命线.本文主要面向交互模块同处一个控制器,且交互模块间需要传输大数据包,如图像等信息的情况进行研究,以共享内存为传输介质,针对传统的共享内存同步机制的不足,提出了基于环形队列的异步传输机制,实现数据的高效传输.通过与当前国际主流的 ROS Sharedmem_transport,LCM 进行对比,证明该设计能够降低数据传输延迟,有效提高数据传输效率,满足大数据包消息的即时传输需求.本系统已在智能无人车实验平台中得到成功实施应用(源码已经根据 BSD 协议开源,可从 https://github.com/Crazycuo/Shared_Memory_IPC 访问).它还可以与现有基于 Sockets 的消息传输机制配合使用.Sockets 适合交互模块运行在同一台或者不同控制器中,且模块间需要传输较小的数据包,如控制命令等的情形.本设计适合交互模块较少且模块间需要传输大数据包如图像等的情形.受共享内存的限制,交互模块必须在同一台控制器中.此外,在传输大数据包时,还可结合对数据进行快速压缩的方法,进一步提高传输的带宽.

在实际的工程运用中,由于单个接受速率较慢的异常模块会成为系统的瓶颈,因此在下一步的研究工作中,将为每个共享内存空间添加异常监测功能,断开异常模块与其连接,并及时释放异常模块占用的数据块,从而保证系统稳定运行.

References:

[1] Ferguson D, Baker C, Likhachev M, Dolan J. A reasoning framework for autonomous urban driving. In: Proc. of the IEEE Intelligent Vehicles Symp. (IV 2008). 2008. 775−780. [doi: 10.1109/IVS.2008.4621247]

[2] Elkady A, Sobh T. Robotics middleware: A comprehensive literature survey and attribute-based bibliography. Journal of Robotics, 2012,2012:Article ID 959013. [doi: 10.1155/2012/959013].

[3] Montemerlo M, Roy N, Thrun S. Perspectives on standardization in mobile robot programming: The Carnegie Mellon navigation (CARMEN) toolkit. Intelligent Robots and Systems, 2003,3:2436−2441. [doi: 10.1109/IROS.2003.1249235]

[4] Gerkey B, Vaughan R, Howard A. The player/stage project: Tools for multi-robot and distributed sensor systems. In: Proc. of the Int'l Conf. on Advanced Robotics. 2003. 317−323.

[5] Quigley M, Gerkey B, Conley K, Faust J, Foote T. Ros: An open-source robot operating system. In: Proc. of the Open-Source Software Workshop of the Int'l Conf. on Robotics and Automation. 2009. 1−6.

[6] Huang A, Olson E, Moore D. Lightweight communications and marshalling for low-latency interprocess communication. Technical Report, MIT-CSAIL-TR-2009-041, CSAIL, 2009.

[7] Li DY, Han W, Zheng SY, Jia P. VirtualSwitch: Distributed inter-process communication in the autonomous vehicle. Patent, 2015104048428, 2015.

[8] Han J, Kim D, Lee M, Sunwoo M. Enhanced road boundary and obstacle detection using a downward-looking LIDAR sensor. IEEE Trans. on Vehicular Technology, 2012,61(3):971−985. [doi: 10.1109/TVT.2012.2182785]

[9] Autonomous car development platform. 2017. http://www.nvidia.cn/object/drive-px-cn.html

[10] Stevens WR, Rago SA. Advanced Programming in the UNIX Environment. 2nd ed., Boston: Addison Wesley Publishing Company, 2005.

[11] Dantam N, Stilman M. Robust and efficient communication for real-time multi-process robot software. In: Proc.of the IEEERAS Int'l Conf. on Humanoid Robots. 2012. 3116−322. [doi: 10.1109/HUMANOIDS.2012.6651538]

[12] Ethzasl_Message_Transport. 2016. http://wiki.ros.org/ethzasl_message_transport

[13] Minix (mini-UNIX) file system. 2013. http://www.linux.org/threads/minix-mini-unix-file-system.4545

[14] Sundell H, Tsigas P. Fast and lock-free concurrent priority queues for multi-thread systems. Parallel and Distributed Computing, 2005,65(5):609−627. [doi: 10.1016/j.jpdc.2004.12.005]

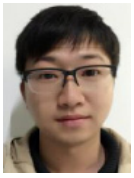
[15] Alexonlinux. 2008. http://www.alexonlinux.com/multithreaded-simple-data-type-access-and-atomic-variables

[16] Ma JG, Huang T, Wang JL, Xu G, Ye D. Underlying techniques for large-scale distributed computing oriented publish/subscribe system. Ruan Jian Xue Bao/Journal of Software, 2006,17(1):134−147 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/134.htm [doi: 10.1360/jos170134]

附中文参考文献:

[7] 李德毅,韩威,郑思仪,贾鹏.智能车分布式进程虚拟交换机通信方法.专利,2015104048428,2015.

[16] 马建刚,黄涛,汪锦岭,徐罡,叶丹.面向大规模分布式计算发布订阅系统核心技术.软件学报,2006,17(1):134−147. http://www.jos.org.cn/1000-9825/17/134.htm [doi: 10.1360/jos170134]



陈存铜(1992－),男,江苏盐城人,硕士,主要研究领域为智能无人车系统架构,通信机制.

邓蓉(1973－),女,博士,讲师,主要研究领域为 Linux/Unix 内核设计、研发与维护,并行分布式系统,CPU+GPU 混合计算平台资源管理与调度.

赵君峤(1983－),男,博士,助理教授,主要研究领域为智能无人车,环境感知与理解,实时定位与建图.

管林挺(1980－),男,博士生,讲师,主要研究领域为基于深度学习的环境感知与理解.

叶晨(1980－),男,博士,高级工程师,CCF 专业会员,主要研究领域为智能无人车,运动规划,行为**决策**.

李德毅(1944－),男,博士,教授,博士生导师,中国工程院院士,CCF 会士,主要研究领域为智能无人车,不确定性人工智能,认知物理学,网络化数据挖掘,互联网计算.