Bernhard Hartmann

Bildverarbeitung in Echtzeit

Bildverarbeitung in Echtzeit unter Windows – dieses Thema bekommt angesichts der angepeilten Steigerungsraten bei der Verwendung von Bildverarbeitungssystemen in der Automatisierung einen immer größeren Stellenwert. Die damit verbundenen Anforderungen sind allerdings erheblich.

Dass Probleme bei der herkömmlichen Bildverarbeitung in Verbindung mit Echtzeit unter Windows sowie den immer schneller und effizienter werdenden Maschinen bestehen, ist den meisten Anwendern klar beziehungsweise hat mancher bereits selbst erfahren müssen. Anhand eines erdachten Beispiels werden nachfolgend verschiedene Lösungswege beziehungsweise Vermeidungsstrategien diskutiert.

Verzögerte Bildaufnahme

Aktuell angebotene Bildverarbeitungssysteme – wie beispielsweise Halcon, Matrox Imaging Library (kurz MIL) oder auch VisionPro – laufen bevorzugt unter Windows. Wie allen Windows-Nutzern bekannt, wurde dieses Betriebssystem für Desktop-Anwendungen entwickelt. Dadurch ergeben sich aufgrund des Windows-Schedulers, der die aktuelle Reihenfolge der Abarbeitung der Aufgaben bestimmt, Verzögerungs- respektive Latenzzeiten, die bis zu 500 ms betragen können. Jeder Anwender kennt das Verhalten, wenn er die Maus bewegen möchte, diese für eine kurze Zeit nicht reagiert und ‚quasi' stillsteht, um sich plötzlich unkoordiniert irgendwohin zu bewegen. Dieses unbeeinflussbare Verhalten kann dazu führen, dass bei der Bildaufnahme – also dem Transport der Bilddaten in den Speicher zum Beispiel über GigE Vision – Verzögerungen auftreten. Wenn nun die Maschinensteuerung beziehungsweise der Prozess darauf angewiesen ist, dass ein Bild zu einer genauen Zeit vorhanden sein muss, kann dies zu einer inakzeptablen Zeitdifferenz führen, die einen Maschinenstillstand zur Folge haben kann.

Ein Beispiel: Beim Öffnen einer Erdnussdose erwartet jeder, dass darin nur genießbare Nüsse enthalten sind. Doch es kommt immer wieder vor, dass man auf Erdnüsse beißt, die sonderbar schmecken (= schlecht). Dies ließe sich schon von der Farbe her erkennen, da die schlechten Nüsse meist wesentlich dunkler sind. Eine Sortierung könnte durch das Herabfallenlassen der Erdnüsse über eine festgelegte Strecke mit gleichzeitiger Bildaufnahme und -auswertung erfolgen. Hierbei darf natürlich kein Bild verloren gehen, eine ungenießbare Nuss muss sicher erkannt und durch Ausblasen innerhalb einer definierten Zeitspanne entfernt werden. Kommt dieses Bild jedoch aufgrund von Latenzzeiten zu spät, ist die Nuss schon zu weit heruntergefallen, um sie noch entfernen zu können.

embeddedworld2016
Halle 4 Stand 640

Komplexe Bildalgorithmen

Die Algorithmen zur Form- oder Farberkennung werden normalerweise auf jedes sich im Windows-Speicher befindliche, eingelesene Bild angewendet. Diese Bildverarbeitungs-Algorithmen können sehr komplex sein und benötigen gegebenenfalls unterschiedlich lange Berechnungszeiten. Dies kann zum Beispiel bei Iterationen auftreten (Funktionen, deren Anzahl der Durchläufe nicht genau vorhersagbar ist). Sollten diese zu lange dauern, lassen sich Abbruchmechanismen definieren. Jeder unter Windows-Kontrolle laufende Algorithmus kann in der Abarbeitung unterbrochen beziehungsweise verzögert werden. Dadurch steht das Ergebnis der

(Bild: Computer&AUTOMATION; Quellen: Matrox, Microsoft, Messe Stuttgart)

Berechnung eventuell nicht zum benötigten Zeitpunkt zur Verfügung, was wiederum zu einer Störung der Maschine führen kann.

Auch hier wieder das Erdnuss-Beispiel: Ideal wäre, dass die Bildaufnahme immer ohne Unterbrechung zeitrichtig erfolgt und kein Bild verloren geht. Die Maschine benötigt zu einer definierten und genau einzuhaltenden Zeit (= deterministisch) ein Signal, mit dem sich das Ausblasventil zur Entfernung der ‚defekten' Erdnuss steuern lässt. Kommt dieses Signal aufgrund der nicht definierbaren Latenzzeiten von Windows zu spät oder gar nicht, wird gegebenenfalls das Ventil geöffnet und eine gute Erdnuss entfernt. Die ‚fehlerhafte' Erdnuss ist inzwischen aufgrund der Verzögerung schon ein paar Zentimeter weiter gefallen oder bereits in der Lieferschachtel angelangt.

Welche Abhilfe beziehungsweise welche Strategien gibt es nun, diesem Dilemma zu begegnen?

Die Framegrabber-Strategie

Eine gängige Lösung sind Framegrabber, auf denen nicht nur die Bildaufnahme, sondern auch die komplette Berechnung direkt vorgenommen werden kann. Hierbei sind die Algorithmen entsprechend vorher als VHDL-Programm zu erstellen, mit dem passenden VHDL-Compiler zu übersetzen – was bis zu einem Tag pro Compilierungs-Vorgang dauern kann – und anschließend zu testen. Dies bedeutet einen hohen Aufwand und zusätzliche Zeit in der Entwicklung. Ohne Programmierung kosten solche Framegrabber einige hundert Euro pro Stück – ohne hierbei das benötigte Interface (GigE, CameraLink oder CoaXPress) beziehungsweise die Kamera mit einbezogen zu haben. Ob dies eine Lösung im Hinblick auf selbstkonfigurierende und autonome Maschinen unter Industrie 4.0 sein kann, muss sich herausstellen.

Eigenentwickelte PCI-Karte

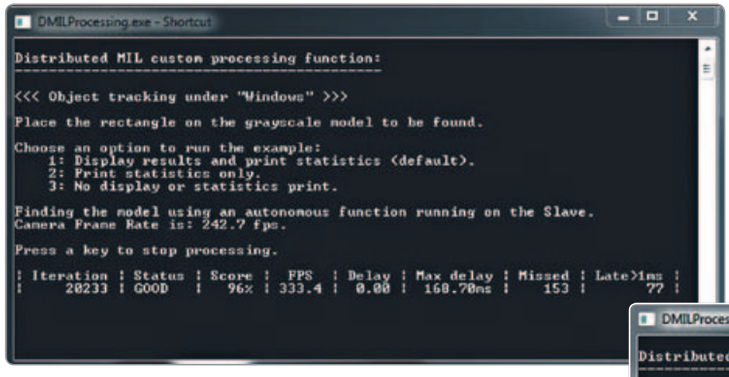
Eine andere Strategie ist die komplette Eigenentwicklung einer PCI-Einsteckkarte auf Basis eines DSP oder eines FPGA. Dies bedeutet, dass die Bild-Erfassung sowie die gesamte Echtzeit-Verarbeitung auf der Einsteckkarte erfolgt und somit auch getrennt von Windows läuft. Allerdings: Das Interface muss selbst programmiert und die Echtzeit-Verarbeitung entwickelt und getestet werden. Zusätzlich ist ein Betriebssystem für den DSP oder für das FPGA nötig, das gegebenenfalls anzupassen ist. Ebenso sind die Bildverarbeitungs-Algorithmen an den jeweils verwendeten Chip anzupassen.

Unvermeidlich ist zudem eine eigene Hardware-Entwicklung mit den zugehörigen Platinen-Entwicklungen, Fehler-Iterationen sowie Test-Szenarien. Und last but not least fällt eine Lagerhaltung für Service- und Ausfälle an. Als problematisch zu bedenken ist zudem eine mögliche Abkündigung eines Bauteils – im ungünstigsten Fall des zentralen, aktiven Bauteils – sowie die Tatsache, dass das Board nur für eine begrenzte Leistung designed ist, mit der Folge, dass ein neues Board entwickelt werden muss, wenn mehr Leistung nötig sein sollte.

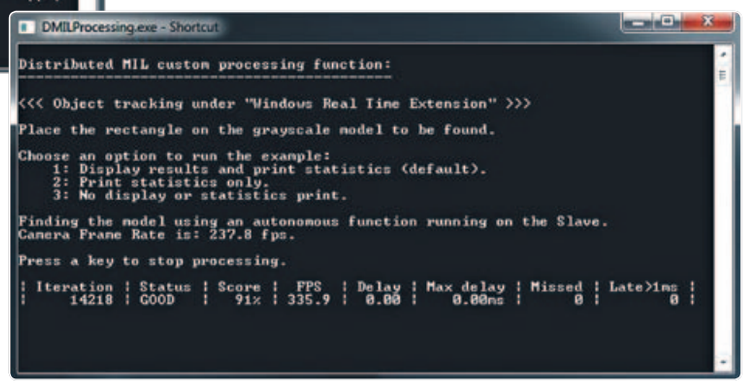
Der Bildverarbeitungs-PC

Aus den beiden Herausforderungen ersannen ‚findige Geister' eine Strategie, die das Problem dem Augenschein nach zwar im ersten Moment löst, sich bei genauerem Hinsehen jedoch als Mogelpackung entpuppt: Die Lösung soll ein eigener PC nur für die Bildverarbeitung sein. An diesem PC darf niemand etwas bewegen, beispielsweise die Maus. In diesem Umfeld gibt es auch Systeme, bei denen verschiedene Windows-Services deaktiviert wurden, um möglichst keine Störung des Bildverarbeitungssystems hervorzurufen. Doch dies führt unter Umständen zu einem instabilen Windows. Und auch mit einem Stand-alone-PC nur für die Bildverarbeitung kann es aus nicht nachvollziehbaren Gründen einige Male am Tag zu einem Bandstopp kommen. Mangels kostengünstiger Alternativen ist diese Strategie allerdings ein weit verbreitetes Szenario in der heutigen Automatisierungswelt.

Um alle Windows-Latenzzeiten in den Griff zu bekommen, wäre es nötig,



Screenshot einer Demo-Applikation, bestehend aus einer Kamera mit 1k × 1k Pixel mit 8 Bit Tiefe, die über ein doppeltes Cameralink-Kabel und den Framegrabber Radient-CL mit 335 fps die Bilder an den PC sendete. Ein Testprogramm (Bildaufnahme und -verarbeitung unter Win7) zeigte speziell bei Bewegung der Maus oder beim Öffnen des Windows-Taskmanagers erhebliche Verzögerungen bei den Berechnungen bis hin zu verlorenen Bildern. ◁



Bei einem RTX64-Subsystems ging kein Bild verloren, auch die geforderte Abweichung <1 ms wurde immer erfüllt. Zusätzlich besteht die Möglichkeit, einen echtzeitfähigen Master-Stack sowie eine selbstprogrammierte Ablaufsteuerung auf dem PC zu integrieren, um auch für zukünftige Anforderungen gerüstet zu sein. ▷

(Bilder: Intervalzero)

dass Bildaufnahme und Bildverarbeitung unabhängig von Windows laufen. Die Bildakquisition müsste, abhängig vom verwendeten Kamera-Interface, als eigenständiger Prozess auf einem eigenen Kern laufen, auf den Windows keinen Zugriff hat. In diesem Fall wären das von der Kamera übertragene Bild beziehungsweise die zugehörigen Bilddaten nicht im Windows-Speicher, sondern in einem von Windows unabhängigen, getrennten Speicher abzulegen.
Ließen sich die Bildalgorithmen zudem unabhängig von Windows auf die Bilddaten anwenden, bestünde die Möglichkeit, mit modernen Mehrkernprozessoren eine komplette Maschine inklusive der Steuerung und der Feldbus-Anbindungen auf einem PC zu implementieren.

Echtzeit-Verarbeitung unter Windows

Eine solche Lösung für die komplette Echtzeit-Bildverarbeitung gibt es von Intervalzero: Sie besteht aus einem PC, einem Cameralink-Framegrabber der Klasse Radient-CL der Firma Matrox mit zugehörigem RTX64-Treiber sowie der Portierung der gesamten MIL-Algorithmen auf das RTX64-Subsystem. Dieses Subsystem kann nachträglich zu Windows installiert werden und ist nach kurzer Konfiguration einsatzbereit. Es fungiert als ‚Symmetric Multi Processing'(SMP)-Erweiterung und transformiert Windows in ein RTOS. Gleichzeitig stellt es die unabhängige Nutzung vorhandener Kerne für die Echtzeit-Anwendungen sicher; ein Kern bleibt immer für Windows reserviert. Sämtliche Mechanismen, wie etwa Mutex oder Semaphore, die ein Software-Entwickler von einem RTOS erwartet, sind vorhanden. Zudem ermöglicht die Entwicklungsumgebung – Microsoft VisualStudio – die Implementierung eigener Algorithmen unter Standard-C/C++. Je nach benötigtem Interface kommen nur Software-Komponenten zum Einsatz, was die Kosten senkt. Anstatt der Kosten für den Framegrabber, der selbstentwickelten Einsteckkarte oder eines separaten PC fallen nur die Lizenzkosten für das linear skalierende Subsystem RTX64 an. Ein PC mit Windows beziehungsweise die MIL-Bibliothek ist in jedem Fall nötig beziehungsweise ist beides oft bereits vorhanden.

Bezugnehmend auf das Beispiel der Erdnuss-Sortiermaschine erfolgt die Bildaufnahme bei diesem Ansatz unter der Kontrolle von RTX64. Somit werden die Bilder ohne Interaktion von Windows in den von RTX64 verwalteten Speicher geschrieben. Da die MIL-Algorithmen auf einem zweiten Core des 4-Kern-Systems laufen und nach erfolgter Bildaufnahme sofort auf das Bild angewendet werden können, könnte die Maschine problemlos die schlechten Erdnüsse während der Fallphase aussortieren. *ik*

Bernhard Hartmann

ist Sales Manager für Zentral-Europa bei Intervalzero in München.

Power hoch 3.

Die neuen VC Z Serien mit ARM Prozessor + VC LINUX® Betriebssystem.

Smart. Leistungsstark. Einfach zu integrieren.

Unsere intelligenten Industriekameras der neuen VC Z Serien überzeugen mit Linux-Betriebssystem, umfangreichen Softwarepaketen und unserem hervorragenden Service. Holen Sie sich die Profis unter den Embedded Systems!

Besuchen Sie uns!
Stand Nr. 1-111

embeddedworld2016
Exhibition&Conference
... it's a smarter world

VC vision components®

WWW.VC-LINUX.COM