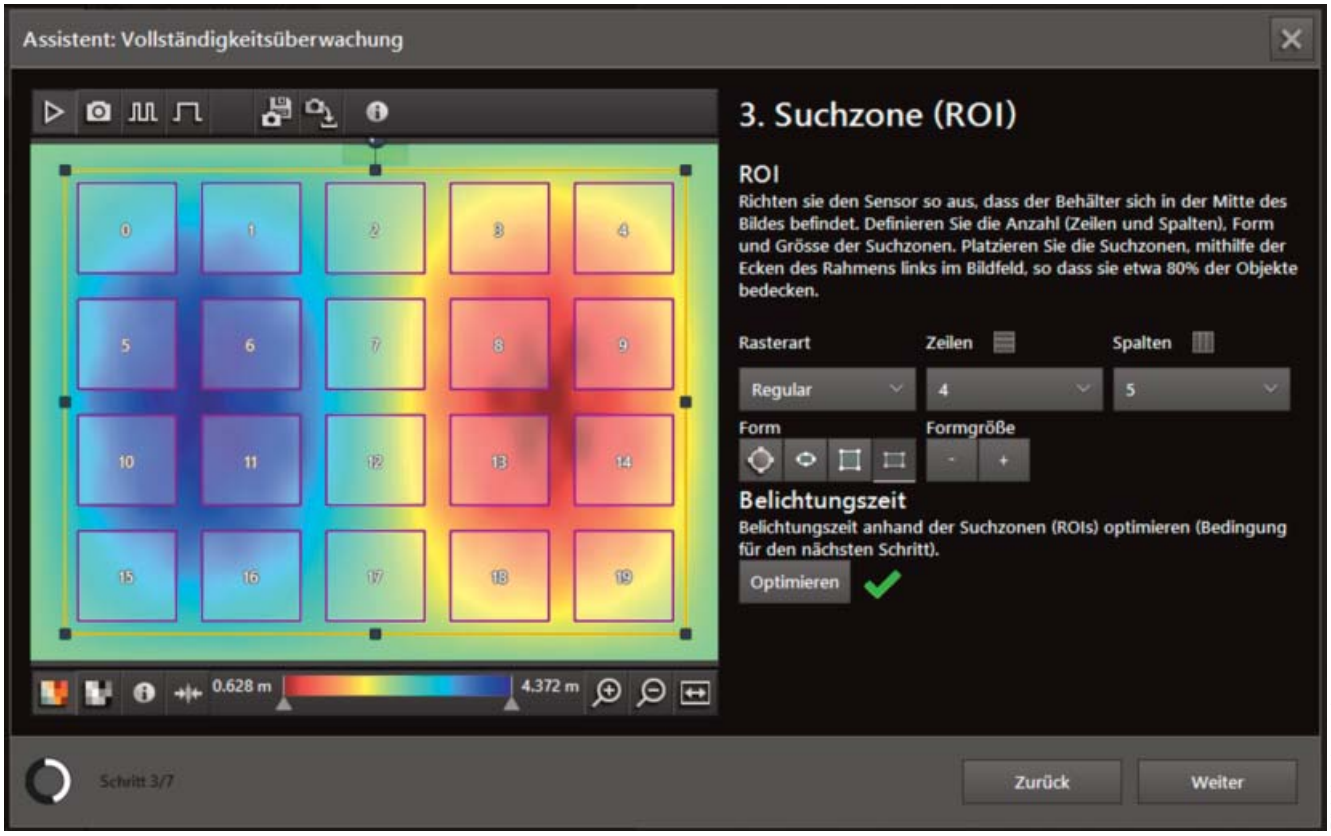

Bild: ifm electronic

Bild 1 | Besser und ansprechender kann man das kaum machen: Eine Seite im Assistent für die Parametrierung einer Vollständigkeitskontrolle in der Software von ifm

Benutzerfreundlichkeit im Fokus

Reportage: Bedienbarkeit von IBV-Software im Test

Immer mehr Anbieter liefern Software für die interaktive Konfiguration von Bildverarbeitungsanwendungen. Bei der Masse an Funktionalität bleibt allerdings schnell die Bedienbarkeit auf der Strecke. Im Rahmen dieses Beitrages sollen die User Interfaces (UI) der Softwarepakete Vision Designer (Cognex), Design Assistant (Matrox Imaging) und Merlic (MVTec) etwas genauer betrachtet werden.

Bildverarbeitungssoftware für die PC-Plattform, die man ohne Programmierung interaktiv für eine industrielle Anwendung konfigurieren kann, gibt es seit Mitte der 90er Jahre am Markt. In den letzten Jahren haben einige der großen Bibliothekshersteller wie Cognex, Matrox und MVTec ihr Produktportfolio um entsprechende Softwareprodukte erweitert. Im Rahmen eines Beratungsprojektes ergab sich die Gelegenheit, neben weiteren Softwarepaketen und Vision-Sensoren den Cognex Vision Designer, den Matrox Design Assistant und MVTecs Merlic genauer kennenzulernen. Alle drei genannten Anbieter positionieren sich ähnlich wie MVTec, die Merlic mit dem Slogan ‘Bildverarbeitungssoftware, mit der Komplettlösungen schnell zusammengestellt werden können, ohne eine einzige Codezeile zu schreiben’ bewerben. Die Vermarktung der Software ist mehr oder weniger offen an die jeweils darunterliegende Bildverarbeitungsbibliothek gekoppelt. Cognex sieht den Designer ‘bundled with VisionPro Software’, für Matrox ist der Design Assistant ‘die grafische Entwicklungsumgebung für die MIL’, lediglich MVTec ist mit Verweisen auf die Halcon-Bibliothek recht zurückhaltend. Allen Softwarepaketen ist gemeinsam, dass sich die Kette der Verarbeitungsschritte einer Prüfanwendung mehr oder weniger graphisch mit der Maus zusammenstellen lässt. Der Anwender baut Schritt für Schritt eine Abfolge von Operationen für die Inspektion eines Prüfteiles interaktiv zusammen. Matrox verwendet als Strukturelement ein stark an die DIN angelehntes Flussdiagramm (Bild 2), Cognex spricht

Auszug aus der inVISION, Machine Vision • Identification • Imaging, Ausgabe 5/2016.

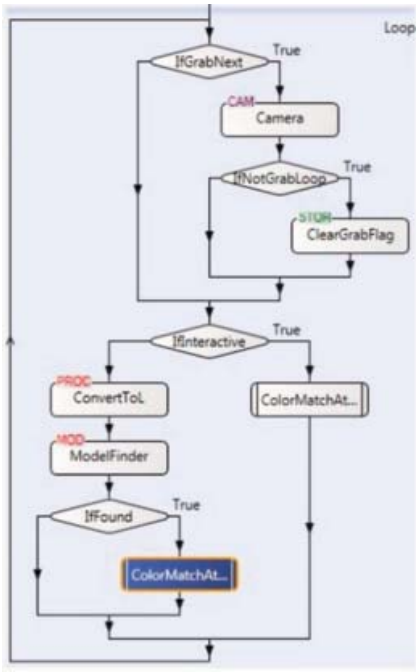

Bild: Matrox Imaging

Bild 2 | Selbsterklärend und übersichtlich: Flowchart im Matrox Design Assistant

von einem ‘graphical sequence diagram’, das gewisse Anleihen sowohl bei Labview als auch Powerpoint erkennen lässt, MVTec hingegen verfolgt einen eigenständigen und optisch sehr ansprechend gestalteten Ansatz, der das von der Kamera generierte Bild in den Mittelpunkt der interaktiven Bearbeitung stellt. Bei Matrox und MVTec konfiguriert man den chronologischen Ablauf der Prüfung im Fenster von oben nach unten, im Cognex Designer hingegen von links nach rechts.

Designer als zentrales Werkzeug

Mit der Bereitstellung der ‘Controldesigner’-Klasse im .NET Framework hat Microsoft bereits 2002 eine Schrittmacher-Funktion für die Kategorie der so genannten graphischen Designer übernommen. Mittlerweile findet sich in vielen Programmen eine daraus abgeleitete Funktionalität, mit der man das Design und die Anordnung von Controls in einem Fenster bequem interaktiv konfigurieren kann. Für die drei hier besprochenen Softwarepakete ist die Designerfunktionalität ein wesentliches Leistungsmerkmal. Alle drei Pakete bieten zusätzlich zur graphisch-interaktiven Konfiguration des Prüfablaufes die Möglichkeit, das Frontend für den automatisierten Prüfbetrieb unter Nutzung graphischer Designerwerkzeuge frei zu gestalten. Insgesamt ist beeindruckend, welchen großen Funktionsumfang die Softwarepakete bieten. Um einigermaßen seriös einen Leistungsvergleich der Tools wie z.B. Pattern Matching, Blob Analysis und Code Reading unter reproduzierbaren Laborbedingungen anzustellen, würde man aufgrund der umfangreichen Funktionalität der hier betrachteten Anwendungen sicher schnell beim Umfang einer Masterarbeit landen. Bei meinen weiteren Betrachtungen möchte ich daher das Thema Algorithmen komplett außen vor lassen. Das

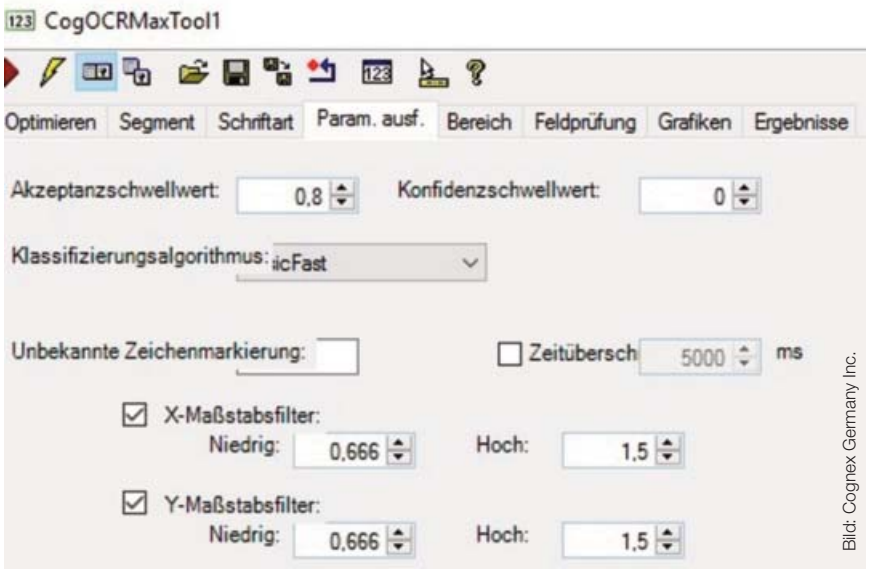

Bild 3 | Lustlos und wenig übersichtlich, dafür wenigstens mit einem gewissen Windows 95 Retro-Charme. Von guter Bedienerführung ist man allerdings weit entfernt.

User Interface (UI) ist die eigentliche Produktneuheit, liegt es doch quasi als neue Schicht über den seit vielen Jahren bewährten Bibliotheken. Zum Thema Usability einer Anwendung möchte ich einige grundsätzliche Überlegungen anstellen.

User Experience als Erfolgsfaktor für Softwareprodukte

Die User Experience (UX) gehört mittlerweile zu den zentralen Begeisterungsfaktoren. Ein schickes User Interface verbunden mit einer möglichst intuitiven Bedienbarkeit ist ein wichtiger Wettbewerbsvorteil. Nachfolgend sind zehn UX-bezogene Einstiegsfragen zu finden, mit denen man sich einer beliebigen Software annähern kann. Spezifische Aspekte der Bildverarbeitung werden dabei berücksichtigt, sie stehen aber nicht im Vordergrund. Zentral ist immer die Frage: Wie ernst meint es der Softwarehersteller mit der Benutzerfreundlichkeit?

UX Feature #1: Dauer der Startprozedur?

Beginn jeder Auseinandersetzung mit einer Software ist der Start der Anwendung. Es ist immer wieder überraschend, wie viel Zeit einzelne Anwendungen mit der Startprozedur vergeuden. Es ist in der Praxis gar nicht so selten, dass eine Software auf einem Rechner mittlerer Leistungsfähigkeit eine Minute und mehr zum Start benötigt. Wenn zehn Mitarbeiter im Unternehmen diese Anwendung fünfmal am Tag laden, dann summiert sich die unproduktive Wartezeit auf über einen Mann-Monat im Jahr. Eine Anwendung sollte nur wenige Sekunden zum Start benötigen:

- <3s: Optimal
- 3 bis 5s: Akzeptabel
- 5 bis 10s: Das Maximum, visuelles Feedback (Fortschrittsbalken o.ä.) vorausgesetzt
- >30s: Katastrophal

Lange Startzeiten lassen sich über einen als Lazy Initialization bezeichneten Implementierungsansatz vermeiden. Im Prinzip geht es darum, Datenobjekte möglichst spät, idealerweise erst zum Zeitpunkt der erstmaligen Verwendung zu laden und zu initialisieren. Ziel ist es, dem Anwender schnellstmöglich das Gefühl der Kontrolle über die Anwendung zu geben. Idealerweise gestaltet man die Initialisierung adaptiv, d.h. man beobachtet und lernt das typische Verhalten des Anwenders und passt die Abläufe der Anwendungsinitialisierung entsprechend automatisch an. Die Programmiersprache Swift von Apple z.B. unterstützt diesen Implementierungsstil über das Schlüsselwort ‘Lazy’ für Objekteigenschaften (Properties). Es soll nicht unerwähnt bleiben, dass es in der industriellen Anwendung durchaus Gründe gibt, über ein gezieltes Pre-Loading möglichst viele Module frühzeitig in den Arbeitsspeicher zu holen. Aber dieser Anwendungsfall ist zumindest bei der Arbeit im Büro bzw. Labor eher die Ausnahme. Idealerweise ist Lazy Initialization konfigurier- und abschaltbar.

UX Feature #2: Freie Skalierung der (Bild-)Anzeige im UI?

Im Bereich der Webentwicklung ist ‘Responsive Design’ aufgrund der Gerätevielfalt auf Anwenderseite seit einiger Zeit das zentrale Thema. Ziel ist es, in Abhängigkeit von bestimmten Eigenschaften des Ausgabemediums, unterschiedliche, auf die Usability optimierte Designs anzuzeigen und das Layout der Controls entsprechend anzupassen. Auch für eine Desktopanwendung ist es interessant zu prüfen, wie das Programm mit unterschiedlichen Bildschirmauflösungen umgehen kann und ob evtl. ein Betrieb an mehreren Monitoren möglich ist. Allen oben erwähnten Softwarepaketen ist gemeinsam, dass der Bildschirm gar nicht groß genug sein kann, um all die für die Konfiguration benötigten Fenster der Entwicklungsumgebung (IDE) darzustellen. Je mehr Flexibilität die Anwendung für die Anordnung und Darstellung der einzelnen Ausgabefenster bietet, desto komfortabler wird das Arbeiten. Warum nicht das Tool-Flow-Fenster separat vom aktuell bearbeiteten Kamerabild auf jeweils zwei HD-Monitoren darstellen? Bilder sollten mitsamt eines optionalen Overlays in der Anzeige stufenlos skalierbar sein. Dazu muss die Anwendung ein modernes vektorbasiertes Renderingmodul verwenden und die Hardwarebeschleunigung der Grafikkarte nutzen.

Auszug aus der inVISION, Machine Vision • Identification • Imaging, Ausgabe 5/2016.

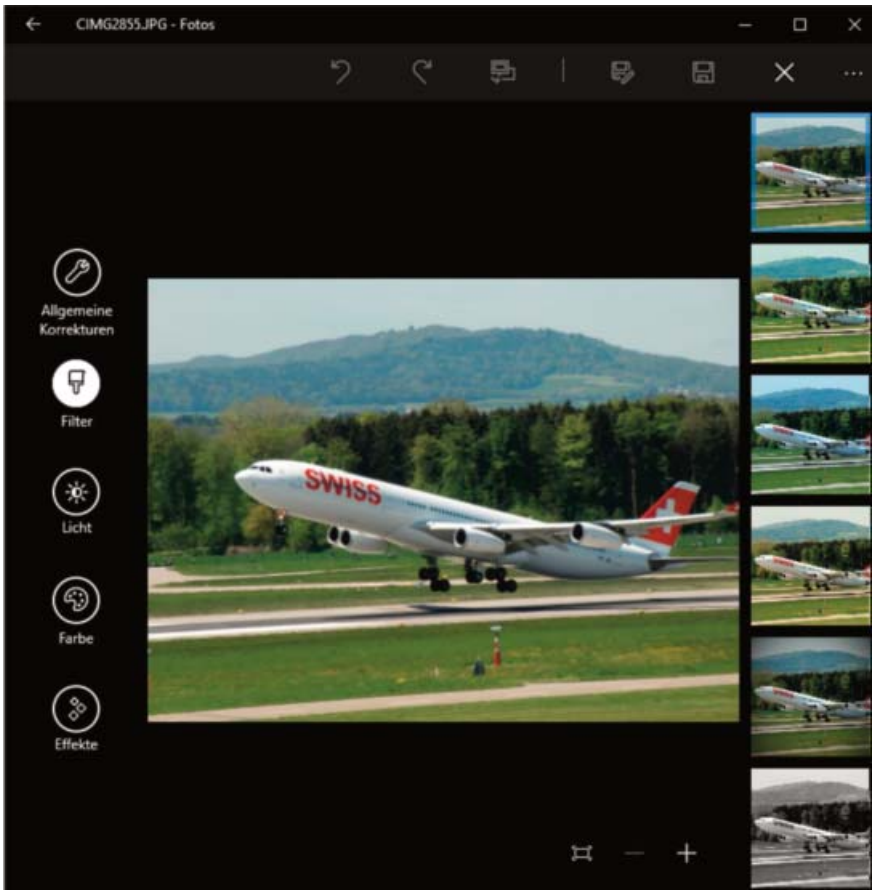

Bild: Microsoft Corporation

Bild 4 | Der Standardbildbetrachter in Windows 10 mit Vorschaufenstern. Selbstverständlich mit Undo und Redo.

UX Feature #3: Ease of use?

Insbesondere im technischen Bereich ist es in vielen Softwareunternehmen ein Problem, dass die Entwickler selten ein Gefühl für Design und Ästhetik haben und auch von der Gedankenwelt und den Kenntnissen der potentiellen Anwender der Software bzw. des Produktes zu weit entfernt sind. Allzu häufig stellt das UI eine 1:1-Ansicht der internen objektorientierten Datenstrukturen dar. Der typische Softwareentwickler fühlt sich mit dieser Vorgehensweise sogar außerordentlich wohl, muss er dadurch doch aus seiner intellektuellen Komfortzone kaum heraustreten. Den Anwender der Software interessiert das herzlich wenig. Er hat ein Problem und benötigt dafür eine softwaretechnische Lösung, die leicht erlernbar und effizient zu nutzen sein muss. Wenn es nur irgendwie geht, am besten durch Ausprobieren, weil man schlicht und einfach keine Zeit und keine Lust für eine intensive Auseinandersetzung mit der Software hat. Damit sind wir erneut beim UI und der Bedienlogik: Einfachheit kann zum entscheidenden Erfolgsfaktor werden. Je besser die Kunden mit einer Software umgehen können und je weniger die Anwender auf die Unterstützung des Herstellers angewiesen sind, desto erfolgreicher lässt sich eine Software einsetzen. Ähnlich groß wie das Angebot an Bildverarbeitungssoftware, intelligenten Kameras und Vision-Sensoren ist auch die Bandbreite der Usability der angebotenen Lösungen. So gibt es äußerst clevere Lösungen, wie z.B. der preisgekrönte intelligente Assistent für die Farbkonfiguration in den Vision-Sensoren der Baumer Verisens XC-Serie. Erstklassig ist auch z.B. die Benutzerführung im Vision Assistant für die Konfiguration der O3D3x Vision-Sensoren von ifm electronic (Bild 1). Sollte ein Designpreis für Benutzeroberflächen in der Bildverarbeitung ausgelobt werden, dann gehört die ifm-Software ganz oben auf die Kandidatenliste. Allerdings ist der Funktionsumfang der Software aktuell noch sehr begrenzt. Ob der Hersteller das hohe Niveau an Usability auch bei der Bereitstellung aufwändigerer Funktionalität wird halten können, muss sich zeigen. Daher sollte man aufpassen nicht Äpfel mit Birnen zu vergleichen. Bedienerfreundlichkeit steht immer im Widerspruch zu maximaler Flexibilität und Leistungsfähigkeit. Während die Vision-Sensoren für eine Teilmenge an Anwendungsgebieten optimiert sind, verfolgen Produkte wie der Vision Designer, der Design Assistant und Merlic einen Universalansatz. Umfangreiche Funktionalität führt allerdings auch schnell zu Komplexität. Zurück zum UI: Die bekannten ‘Eight Golden Rules of Interface Design’ des US-amerikanischen Informatikprofessors Ben Shneiderman fordern u.a. das Prinzip der Abgeschlossenheit: Aktionssequenzen sollten in Gruppen organisiert sein, die einen Anfang, einen Mittelteil und ein Ende aufweisen. Aus meiner Sicht sind daher zu viele gleichzeitig geöffnete nicht-modale Navigations- und Parametrierfenster, wie man sie in den hier vorgestellten Softwareprodukten stellenweise findet, eher problematisch. Schwer zu durchschauen sind auch Property-Dialoge mit einer großen Zahl an Property Pages, die untereinander logische Abhängigkeiten aufweisen. Ein Parameterdialog wie der in Bild 3 ist kaum bedienbar. Hier wurde eine Vielzahl an Grundregeln für die Gestaltung von Dialogen von den Entwicklern missachtet: Viel zu viele Tab Pages (insgesamt acht), Parameter (z.B. Schriftart, Bereich) neben Aktionen (Optimieren) und Ergebnisausgaben, diverse wenig selbsterklärende Icons in der Toolbar, Controls und Textlabels ohne Ordnung, zum Teil überlappend. Selbst wenn ein

Anwender sich die Funktionalität dieses Dialogfensters mühsam erarbeitet, wird er nach einer Pause von wenigen Wochen erneut über Anordnung und Logik der Seiten rätseln.

UX Feature #4: mehrstufiges Rückgängig/Wiederherstellen?

'Permit easy reversal of actions' ist eine weitere Forderung innerhalb der goldenen Regeln. Anwender möchten sich die Funktionalität einer Anwendung spielerisch erarbeiten können. Dazu gehört die Sicherheit, vorgenommene Eingaben und Änderungen auch problemlos wieder rückgängig machen zu können. Eine benutzerfreundliche Anwendung bietet daher eine mehrstufige Undo- und Redo-Funktionalität. Die Implementierung einer solchen Funktionalität ist nicht trivial und muss bei der Softwareentwicklung frühzeitig berücksichtigt werden. Mehrstufiges Undo/Redo erfordert in der Regel die Umsetzung einer sogenannten Event-Sourcing-Architektur, die für die Entwickler so manche Herausforderung bereithält. Merlic bietet immerhin ein einstufiges Undo/Redo, die anderen Softwarepakete müssen hier allerdings passen.

UX Feature #5: Reaktiv durch Parallelisierung?

Seit über zehn Jahren sind Multi-Core CPUs für den PC-Massenmarkt verfügbar und auch im embedded-Bereich finden Multi-Core-Technologien immer mehr Verbreitung. Eine moderne Anwendung sollte daher die Leistungsfähigkeit der im Rechner vorhandenen Hardware (CPU und GPU) möglichst vielschichtig ausnutzen. Dazu gehört einerseits die parallele Verarbeitung von Bilddaten, andererseits aber auch der überlegte Einsatz von Nebenläufigkeit im UI. Die Verteilung eines (geeigneten) Bildverarbeitungsalgorithmus auf mehrere Threads ist mit moderner Compiler-Unterstützung heutzutage nicht mehr die große Herausforderung und sollte daher im Bereich Bildbearbeitung und Bildverarbeitung

Bild: Fraunhofer IPA

Bild 5 | Ein Umbruch wie schon lange nicht mehr: Industrie 4.0 verändert die ITK-Branche.

Standard sein. Anders sieht es bei der Aufgabe aus, mithilfe von Multi-Threading das UI möglichst reaktiv (reaktionsfähig) zu halten. Dauerhaft abgedimmte und eingefrorene Fenster wirken immer unprofessionell und sind für den Anwender wenig vertrauenerweckend. Der Anwender möchte ohne Wartezeit jederzeit eingreifen können und längere Berechnungsprozesse durchaus aktiv abbrechen. Ohne eine saubere Anwendungs-Architektur lässt sich derlei Komfort kaum stabil implementieren.

UX Feature #6: Erweiterbarkeit?

Erweiterbarkeit bezeichnet hier nicht Änderungen an Datenstrukturen und Programmaufbau auf Seiten des Softwareherstellers im Rahmen der üblichen Wartungsphase der Software. Unter Erweiterbarkeit versteht man die Option für den Anwender der Software, ohne Beteiligung des Herstellers die Software über öffentlich dokumentierte Schnittstellen (APIs) funktional zu erweitern und an eigene Bedürfnisse anzupassen. Insbesondere im industriellen Umfeld schafft eine Plug-in-Schnittstelle die Freiheit, z.B. proprietäre Geräteanbindungen und Kommunikationsprotokolle eigenständig nachzurüsten. Bei vergleichbarer Funktionalität variieren die von den Anbietern verwendeten Bezeichner: Matrox z.B. nennt Plug-ins 'custom steps', Cognex spricht von 'custom plugins' und MVTec von 'custom tools'. Große Softwaresysteme werden durch das Vorhandensein leistungsfähiger Erweiterungsschnittstellen besser wartbar. Microsoft z.B. unterstützt derlei Ansätze mit dem Managed Extensibility Framework (MEF) und stellt für die Implementierung von Plug-in-Mechanismen die benötigte Infrastruktur (einschließlich einer optionalen Lazy Initialization) bereit. Im Rahmen einer so genannten dynamic-discovery-Phase wird zum Start der Anwendung in einem vordefinierten Pfad nach Modulen bzw. Assemblies gesucht, die ihre Funktionalität entsprechend vordefinierter Regeln exportieren. Diese Funktionalität steht dann in der Anwendung zur Laufzeit zur Verfügung. Für den Anwender einer Software ist es vorteilhaft, wenn sich ein umfangreicher Drittmarkt entwickeln kann, weil die technischen und kommerziellen Einstiegshürden für die Implementierung von Plug-ins niedrig sind. Idealerweise unterstützt der Softwarehersteller die Entwickler von Plug-ins, indem er zwischen Anbieter und möglichen Abnehmern proaktiv vermittelt.

UX Feature #7: Gerätesimulation?

Die Anbindung von (Kamera-) Sensorik und die Kommunikation mit (Roboter-)

Steuerungen und übergeordneten Leitrechnern sind wesentliche Leistungsmerkmale eines Bildverarbeitungssystems. Bei der Arbeit im Büro steht die Prozessperipherie jedoch üblicherweise nicht zur Verfügung. Die Software sollte daher flexible Mechanismen bieten, die Konfiguration und Optimierung des Systems auch ohne Vorhandensein der Prozessperipherie durchführen zu können. Im Idealfall lassen sich die unterschiedlichsten Geräte innerhalb der Software mit wenig Konfigurationsaufwand transparent simulieren. Datenobjekte, die innerhalb der Software im Rahmen der Simulation bearbeitet wurden, sollten 1:1 im realen System verwendet werden können und umgedreht.

UX Feature #8: 64Bit?

Mit Speicherproblemen möchte sich heutzutage kein Anwender mehr herumärgern müssen. Bildverarbeitungsapplikationen sind dafür berüchtigt, dass sie einen großen Speicherbedarf haben können. Es kann durchaus sein, dass z.B. der Pattern-Matching-Algorithmus im Hintergrund mehrere Kopien des aktuell bearbeiteten Bildes hält. Auf einer 64Bit-Plattform sollte die Software daher unbedingt als native 64Bit-Anwendung verfügbar sein. Ausreichend Arbeitsspeicher ist hilfreich für die Darstellung einer Echtzeitvorschau, so wie man sie in Bildbearbeitungssoftware für Fotokorrektur und Fotomontage findet. Der Standardbildbetrachter von Windows 10 (Bild 4) zeigt stilvoll, wie man mit aussagekräftigen Icons, einem Minimum an Text und ein paar Preview-Fenstern eine intuitive Bedienbarkeit erzielen kann. Leider findet man solche Vorschaufenster viel zu selten in den Parameterdialogen der hier betrachteten Bildverarbeitungssoftware.

UX Feature #9: Lokalisierung des UI?

Wer mit der Software auf internationale Märkte geht, muss die Anwendung in mehreren Sprachversionen liefern können. Merlic hat mit aktuell neun Sprachversionen hier ganz klar am meisten zu bieten. Dass Texte in unterschiedlicher Sprache bei der Anzeige verschieden lang sein können, muss vom Hersteller berücksichtigt und getestet werden. Darstellungsfehler wie im Dialog in Bild 3 sind zu vermeiden.

Die zukünftige Lokalisierung des UI muss von den Entwicklern der Software zu einem frühen Zeitpunkt mitberücksichtigt werden. Idealerweise sind Texte, Icons und Bilder vollkommen separiert vom Rest der Implementierung und können auch extern bearbeitet bzw. weiterverarbeitet werden.

UX Feature #10: Fehlerbehandlung?

Eine gute Fehlerbehandlung verbunden mit aussagekräftigen Fehlermeldungen ist eine wichtige Funktionalität jeder Software. Viele Anwender erarbeiten sich eine Software iterativ über Ausprobieren, daher kommt der Qualität von Fehlermeldungen eine große Bedeutung zu. Zu einer Fehlermeldung gehört bei den üblichen nicht-trivialen Anwendungen im industriellen Umfeld immer auch eine eindeutige Fehlernummer. Eine Fehlernummer schafft Klarheit in der Kommunikation mit dem Hersteller der Software und vermeidet Missverständnisse im Kontakt mit dem technischen Support. Die Fehlerbehandlung einer Bildverarbeitungssoftware lässt sich immer gut testen, indem man mitten im laufenden Betrieb die Verbindung zu angeschlossenen Peripheriegeräten (Sensorik, Kamera, Prozesskommunikation) durch Abziehen des Kabels unterbricht, auch wenn die meisten Vertriebsingenieure über derlei Ansinnen selten begeistert sind. Probieren Sie es trotzdem! Spannend ist, in welcher Form die Software diesen Vorgang im UI protokolliert und welche Handlungsempfehlungen ausgesprochen werden. Man findet leider auch immer wieder Software, die beim Kappen der Verbindung instabil wird oder sich sogar schlicht aufhängt. Ein erneutes Einstecken des Kabels sollte das System wieder in den ursprünglich stabilen Zustand zurückversetzen.

Fazit

Die Hersteller der hier betrachteten Programme haben sich der großen Herausforderung gestellt, ihre umfangreichen Algorithmensammlungen mit einem Framework zu versehen, das einem mehr oder weniger qualifizierten Anwender ohne Programmierkenntnisse die Konfiguration einer industrietauglichen Bildverarbeitungslösung ermöglichen soll. Die Implementierung solch einer Software für die typischen Qualitätsanforderungen im industriellen 24/7-Umfeld ist kein Kinderspiel und dürfte über einen längeren Zeitraum beträchtliche Entwicklungsressourcen gebunden haben und auch weiterhin für die Fortentwicklung und Wartung binden. Alle drei Hersteller kämpfen bei der Konzeption der Software mit dem Problem, wie weit man Funktionalität über eine interaktive Konfiguration zugängig macht und ab welchem Punkt man verlangt, dass die erforderliche Logik eben doch in irgendeiner Form ‘programmiert’ wird? Insbesondere Cognex bietet hier über Skripte, die man in C# innerhalb der IDE direkt eintippen kann, einen sehr flexiblen Ansatz mit teils beängstigendem Durchgriff bis in die Tiefen des Systems. Anstelle quickly build ist dann aber wieder sorgfältig Nachdenken, Programmieren, Dokumentieren, Versionieren und Archivieren angesagt. Der für diese Produktkategorie zentrale Claim einer interaktiven Konfiguration wird dabei dann natürlich auf den Kopf gestellt. Als Anwender muss man sich immer bewusst sein, dass es sich jeweils um proprietäre Insellösungen handelt. Wer die Leistungsfähigkeit der Software einigermaßen ausreizen will, muss mit beträchtlichen Anfangsinvestitionen in die Qualifizierung der Mitarbeiter rechnen. Dieses Investment geht bei einem Anbieterwechsel zu einem erheblichen Teil verloren, auch wenn sich die Konzepte der Softwarepakete in vielen Punkten ähneln. Dennoch wird es spannend zu beobachten sein, in welchem Umfang diese All-in-one-Softwareprodukte am Markt in den nächsten Jahren Verbreitung finden. Der zunehmende Fachkräfte-Mangel in der Industrie spricht ganz klar für Programme wie Merlic & Co. Wenn Personal knapp ist, geht es nur noch darum mit den vorhandenen Ressourcen in möglichst kurzer Zeit viel Output zu erzielen. Andererseits scheuen erfolgreiche Integratoren zuweit gehende Abhängigkeiten und möchten möglichst viel Kontrolle über die eigenen Systeme behalten. Viele werden sich auch zukünftig auf der sicheren Seite sehen, wenn sie das Framework für die eigenen Bedürfnisse zugeschnitten selbst implementieren und wie bisher lediglich die Algorithmen zukaufen oder sich dabei kostengünstig der zunehmend vorhandenen Open-Source-Angebote bedienen.

Ausblick

Während sich ein Großteil der Informations- und Kommunikationstechnologie-Branche mitten in einem umfangreichen Paradigmenwechsel sieht (Bild 5) und mit skalierbaren und leichtgewichtigen, oftmals cloudbasierten Diensten völlig neue Lösungsansätze und Lizensierungsmodelle entwickelt, scheinen die Softwareanbieter in der Bildverarbeitung an traditionellen Anwendungsarchitekturen und Implementierungsverfahren festzuhalten. Etwas revolutionär Neues konnte jedenfalls bei der kürzlich vorgenommenen Marktevaluierung nicht entdeckt werden. Die Liste der von der Software bereitgestellten Funktionalität verlängert sich kontinuierlich und Gutes wird detailversessen immer weiter verbessert. So entstehen sukzessive immer umfangreichere monolithische Lösungen, die aber kaum noch zu überschauen sind und zunehmend schwerfällig wirken. Ich bin überzeugt, dass sich auch in der Bildverarbeitung Softwarehersteller mit dem grundlegenden Umbruch eingehend beschäftigen und sowohl ihr Angebot als auch ihr Geschäftsmodell radikal überdenken. Wenn große IT-Konzerne wie Google, Facebook und IBM Bildverarbeitungsfunktionalität und künstliche Intelligenz als Cloudservices anbieten und zugleich zunehmend Code aus ihren Labors der Entwickler-Community als Open Source bereitstellen, dann wird das auch Auswirkungen auf die mittelständisch strukturierte Bildverarbeitungsbranche haben. ■

www.softwareforindustry40.de

Autor | Christian Demant, Consultant

Auszug aus der inVISION, Machine Vision • Identification • Imaging, Ausgabe 5/2016.