Using IWR to Cut Labor Costs Without Outsourcing

An Industry Report on Intelligent Word Recognition
by Arthur Gingrande, ICP Partner

imerge CONSULTING

Intelligent word recognition (IWR) technology lies at the heart of A2iA FieldReader software. This report explains intelligent word recognition (IWR) technology and how it differs from ICR, enumerates the data entry applications best-suited for IWR, and describes the technology's major benefits.

IWR is optimized for processing field-level (as opposed to character-level) data on real-world documents that contain mostly free-form, hard-to-recognize handwriting that is inherently unsuitable for ICR. The best use of IWR is to eliminate a high percentage of the manual entry of handwritten data and run-on hand print fields on documents that otherwise could be keyed only by humans.

The IWR process of converting a written word into computer-usable data occurs in a series of stages that when combined with the use of a dictionary of terms produces the highest IWR accuracy. With IWR, the cost of an error is critical, depending on how far downstream in the data entry process the error is corrected. This report analyzes several methods of error reduction using IWR and shows their costs.

IWR tools such as confidence thresholds that calibrate accuracy are explained. They provide an IWR vendor with a rational means of pricing their software economically and fairly. Of course, some applications, such as recognizing "open" handwritten fields in surveys and traffic tickets, are better suited than others for IWR deployment. The major A2iA FieldReader applications - how they are defined and what the payoff is - are listed and described.

IWR provides a competitive advantage for operations that rely on the conversion of paperwork into computer-usable form, such as data entry departments of big government agencies, large financial institutions, and service bureaus. In fact, IWR can automate data entry at a cost so low that it enables A2iA FieldReader customers to successfully compete against offshore service bureaus located in countries such as India and China.

www.a2ia.com I 917.237.0390 I marketing@a2ia.com

Introduction: the Challenge of Reducing Manual Data-Entry

As corporate America nears the end of its transformation from the Industrial Era into the Information Era, the information explosion continues to fuel a big demand for data processing labor. Every day, America's key entry workforce is collectively responsible for indexing or converting over one billion pages of paper-based information into computer usable ASCII data, which they accomplish with consistently high accuracy and rapid throughput.

According to the year 2000-2001 U.S. Department of Labor publication, Occupational Outlook Handbook (http://www.umsl.edu/services/govdocs/ooh20002001/313.htm), workers in the information processing industry, approximately806,000 workers held jobs in 2000 and were employed in every sector of the economy. Approximately 509,000 were data entry operators and 297,000 were word processors and typists.

According to the 2006-2007 edition of the U.S. Department of Labor publication of the same name (http://www.bls.gov/oco/ocos155.htm#nature), there were 281,000 fewer workers in the industry in 2004 than there were in the year 2000. About 525,000 of them held jobs in all vertical market segments. Nearly 330,000 were data entry operators and 195,000 were word processors. Some workers were telecommuters, keying from home on PCs linked to the main office by dialup modems or broadband connections. The majority that left data entry positions were high-speed key entry professionals who retired or changed jobs.

Why such a dramatic decline in the number of professional key operators over those past five years? In all likelihood, there were two causes for the drop-off in employment: (1) the growing tendency to outsource data entry tasks offshore; and (2) the mainstream adoption of forms-centric, character recognition software that automates data entry tasks. Known as intelligent character recognition (ICR) software, this technology converts images of hand print fields on forms into computer-usable data. In other words, ICR replaces keystroke labor, thereby reducing the need for data entry operators. After more than a decade, data entry managers have successfully integrated ICR into their everyday operations. Because it does not require a high skill level to key rejected character fields, this means that ICR is replacing the fastest and most experienced key operators while slower and lower-paid workers are being hired to key in the fields of character rejects.

As recognition technology continues its decade long march toward mainstream adoption, the fundamental metrics of data entry productivity are changing, and with them the economics of forms processing and other data-intensive operations are changing as well. Improved recognition accuracy is the primary factor that enhances productivity within an automated forms processing environment. Continual progress in recognition accuracy and hardware development, driven by Moore's Law, has advanced ICR technology to the point where, typically, 60% of data entry labor costs can be eliminated by using ICR software.

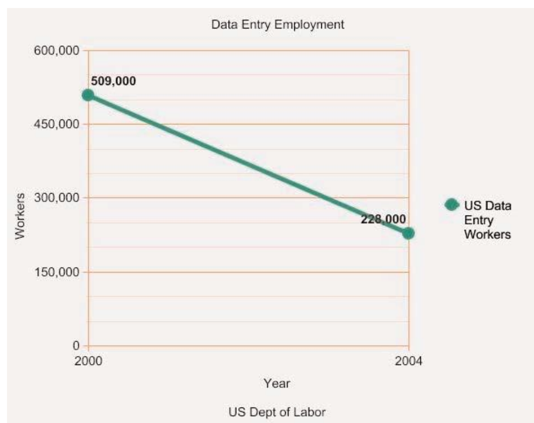


Figure 1

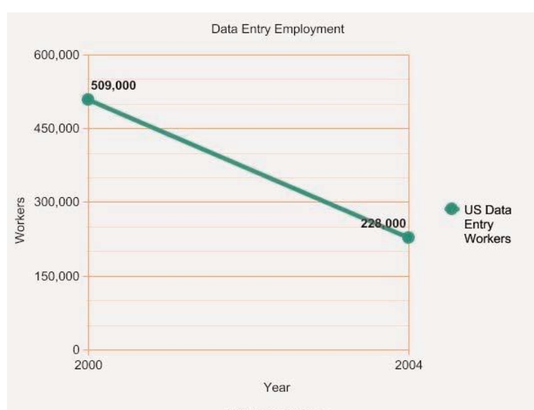


Figure 2

When a field-tested, high technology application produces hardcore cost-savings, the growing competition invariably forces software developers to push that technology to its known limits. Recently, the document processing marketplace has seen the introduction of A2iA's FieldReader, a cutting-edge software application driven by intelligent word recognition (IWR). The development of IWR software might well represent a major evolutionary step in recognition technology. This report explains IWR technology and how it differs from ICR, enumerates the data entry applications best-suited for IWR implementation, and describes its benefits.

How IWR Technology Works

IWR technology lies at the heart of all A2iA's recognition software products. It is best explained by differentiating it from its forerunner, ICR. IWR differs from ICR primarily in that conventional ICR technology recognizes data fundamentally at the character level, while IWR recognizes data at the word or "field" level. A2iA's IWR engine is capable, in fact, of extracting all types of field-based information from a form - either constrained (machine print, hand-printed capitals) or unconstrained (freeform hand print, cursive) from virtually any type of document. So, in many respects,
IWR technology is more evolved than hand print ICR.

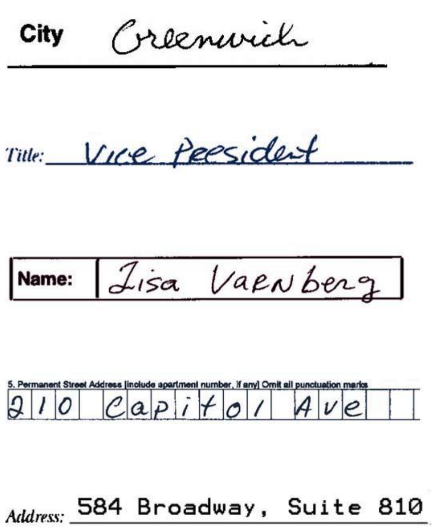


Figure 3

Nevertheless, IWR is not meant to replace conventional ICR and OCR systems, because there are certain applications for which character-based recognition systems are perfectly suited. For example, ICR does an excellent job of recognizing, with extremely high accuracy, characters printed on custom-designed, "ICR-friendly" forms that make use of special inks and graphical devices - such as combs, boxes, or tick marks - that segment hand-printed words on a form into individual characters and enable efficient data extraction from form images. In contrast, IWR is optimized for processing real-world documents that contain mostly free-form, hard-to-recognize data fields that are inherently unsuitable for ICR. This means that the highest and best use of IWR is to eliminate a high percentage of the manual entry of handwritten data and run-on hand print fields on documents that otherwise could be keyed only by humans. In other words, IWR often complements ICR and competes very effectively against cheap labor - either the domestic or the offshore variety.

In its natural state, cursive handwriting (and run-on hand print, which is often found mixed with cursive handwriting) appears unconstrained; not infrequently, individual words are composed of multiple fragments, and each is composed of several handwritten characters joined together. As a result, the bottom-up approach favored by hand print ICR - where words are broken down into single characters and classified, then assembled back into words - exhibits a high failure rate when applied to the recognition of handwritten words. The reason for this has much to do with the way people think when they form cursive words versus when they print words letter-by-letter. Although handwritten words are constructed as strings of individual characters, the writer is conceiving them and forming them at the level of a word or phrase, or even a sentence. This removes the incentive to write individual characters legibly - so long as the overall shapes of the words are intelligible. That is why people can routinely decipher handwriting that obscures the clarity of individual letters: because by using context, the mind is able to recognize words intuitively, without first dissecting them into discrete characters.

Just as a people can only recognize the meaning of words that they already know, so IWR can only work in situations where only a determinative set of word possibilities can appear in a field or document. For example, when IWR is used to read a field in an insurance application form that asks for the make of a car, IWR can only select a word from the dictionary. The field will be filled with words like "Ford" or "Volvo" or "MG." These words must be included in a field-specific dictionary or "field lexis", because IWR can only return results contained within it.

Accordingly, IWR employs an approach toward handwriting recognition that is top-down: it first interprets the overall shape of a word or phrase, and then drills down and classifies more fundamental levels of cursive elements. During the process, the IWR engine concurrently employs different classification techniques that run in parallel on manifold levels, and then it puts them all together and matches the results against a dictionary of word possibilities. So IWR represents an intuitive, holistic, multi-layered approach to recognition, usually driven by neural network technology.

The Anatomy of IWR

The IWR process of converting a written word into computer-usable data occurs in five major stages. Not all of the stages that follow take place serially; a number of them are performed concurrently, in parallel. Collectively, they make up the process referred to as intelligent word recognition. A brief description of each stage is described below.

1) *Image Preprocessing*: During the preprocessing stage, the document image is cleaned up. Handwritten field images are enhanced using sophisticated image enhancement techniques. The document is classified and the document image is registered and deskewed, so that the fields designated for recognition can be efficiently located. The IWR engine identifies the location of the individual data fields on the form image that require recognition. During image cleanup, the objective of the IWR software is to remove "noise" - such as underlines, combs, boxes or stray marks - from the data fields that might be mistaken for handwriting. Additionally, the form image may contain fine print, and other "passive" form attributes that must be separated from the form image, so that only the "active" handwritten data is left behind.

2) Normalization: Once the individual handwriting image fields have been located and cleaned up, the handwriting is normalized. During normalization, each handwriting field image is individually de-slanted and de-skewed in order to remove the variability from the individual handwriting styles. This corrective process enables the handwriting fields to be more easily fitted to a cursive paradigm to which IWR can apply a variety of classification algorithms.

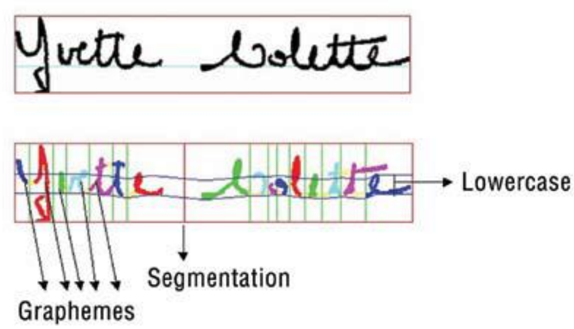


Figure 4

3) *Segmentation*: The IWR engine segments the handwritten fields into separate words (if necessary) and each word further is broken down into individual (or parts of) letter pieces called graphemes. During classification, the sequence of graphemes is examined for various combinations that can be compared with a vocabulary of words associated with a particular field that represent the universe of possible match candidates.

4) Feature Extraction: During this stage, the shape of the word field is analyzed through various techniques of dimensionality reduction and spatial mapping. At the word shape level, forgiveness for illegibility is high. In addition to other analytical techniques, IWR invokes ascender/descender analysis, a process that examines the gross shape of a word or a phrase and then compares the result with a library of patterns that represent upstrokes and downstrokes of the word(s) relative to an imaginary line drawn through the horizontal axis of the verbal field. The longer the word field is, the more idiosyncratic and distinguishable the word shape is, and the greater its value is in the word classification process.

5) Classification: Classification entails a complex synthesis of manifold data collected through different modes of comparative analysis and constellated around a word field. Various combinations and permutations of grapheme data and word shape data are weighed against each other and matched up with entries in word lists and in dictionaries of aliases, and each of them is assigned a confidence value by a neural network based classifier, which ranks each one by confidence value and then exports them to the IWR application in question.

From Graphemes to Words

As briefly mentioned earlier, IWR employs classification models used to interpret a set of data elements called graphemes. A grapheme is the atomic unit or irreducible graphic symbol out of which a character, a letter, a number, or a letter combination can be constructed and then transformed into verbal units such as words. A grapheme as a basic building block of handwriting into which ligatured word shapes can be broken down, and then reconstructed. A grapheme is a shape that could represent, say, a piece of letter or letter combination, as shown in figure 4. Every grapheme is described by a vector that contains statistics about it (average number of black pixels, presence of a loop, etc.). Vectors are processed by mathematical models that analyze the graphemes and their sequences. Because cursive handwriting is highly varied, different grapheme permutations can represent the same letter combination. Multiple neural nets can be used to differentiate between grapheme combinations with low confusion tolerances.

At the data field object level, combinations of words or phrases may be identified, depending upon their content. Classification takes place through the use of grammatical and syntactical rules in conjunction with an application-specific dictionary. Just as an English dictionary uses phonetic symbols that represent individual sounds (phonemes) to give users guidance on how to pronounce a word, the IWR engine has its own internal system of keys, each of which describes a pattern of graphemes that correlate with each word in its specific dictionary.

The IWR engine calculates a confidence value for every word that it compares with a customized vocabulary list, and the results are used to determine the word(s) most likely to be inscribed in the image. For each word analyzed, the system considers various shape and grapheme groupings in order to calculate the confidence value associated with the word(s) in question.

Creating an IWR Dictionary

An IWR dictionary is an inclusive set of words that can be recognized within a field. Its vocabulary can contain anything from a few words (yes, no, maybe) to several thousand words such as a dictionary of USA city names. Building it requires listing nearly all of the possible responses that could be anticipated to be written in the field. Right-sizing is the key to building a word list for IWR. Too many words will create too many possibilities and lower confidence, and too few words in the dictionary will create many rejects, where the word cannot be recognized. A field lexis must contain all of the words as well as all of their aliases. In the terminology of IWR, an alias is a word or abbreviation that has the same meaning as another word or abbreviation, and is thus interchangeable with it. For example, if the vocabulary of an IWR application is limited by a manageable number of terms, then a list of aliases permits each word or phrase to be identified according to its overall shape independent of the classification of the individual characters that it contains. This can be done because there are only so many ways that the make and model of a car written on a traffic ticket can be expressed in verbal form - e.g., "Cadillac Seville", "Caddy Seville", "Cad. Sev.", and "Cadillac S." - and each one of these aliases has an idiosyncratic written shape.

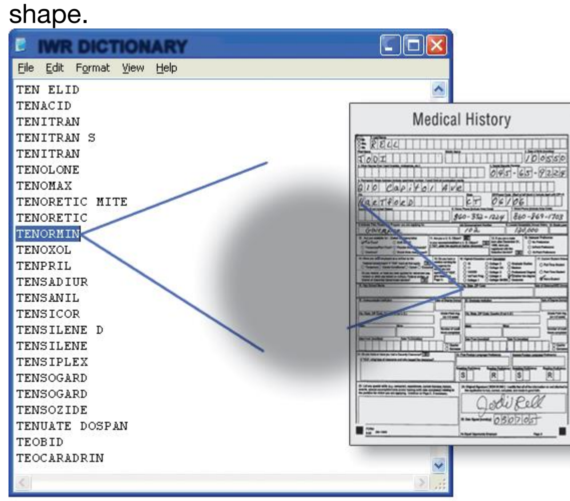


Figure 5
Custom dictionaries containing thousands of words help achieve the highest accuracy.

The use of a dictionary produces the highest IWR accuracy because it ensures that there are no mistakes in spelling at the field-level, which is where the more obvious errors usually occur. In a typical ICR application, one low-confidence character can be responsible for rejecting an entire field. In an IWR application, however, verification and correction are faster and more accurate because validation occurs at the field level and a dictionary can easily correct spelling errors when the overall field-level confidence value is high.

Confidence Thresholds & Recognition Accuracy

The greatest concern of the I.T. department head or the data entry manager who wishes to substitute IWR for data entry labor is the quality of recognition accuracy. Of course, 100 percent accuracy is difficult to obtain under any circumstances - even by the most highly skilled data entry operator. In fact, human accuracy is defined by professional membership organizations such as TAWPI* as 99.5% per character, the number that establishes the gold standard for ICR.

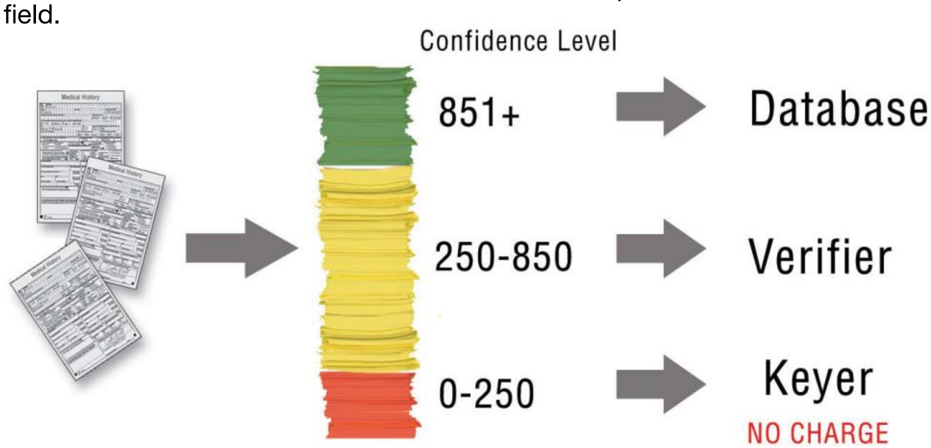
What does this mean? It means that if a professional data entry operator were to key 10,000 characters in one hour, then an average of 9,950 (99.5 x 10,000) characters would be entered correctly and the remaining 50 characters would be entered incorrectly. Based upon a 99.5% per-character accuracy rate, field-level accuracy is a factorial computed by multiplying the accuracy rate times itself times the number of characters in the field, expressed by the formula: FA = 99.5n, where FA = field accuracy and n = the number of characters in the field.

For example, to find the overall human accuracy rate of a five-character field, the value is a product that is calculated as follows: .995 x .995 x .995 x .995 x .995 = .975 or 97.5%, which is the field-level accuracy value for a five-character field. The same kind of calculation can be applied to ICR and IWR data to obtain their equivalent field-level accuracy rates.

Classifying the characters in a data field, however, is only half the story when it comes to accuracy. In order to get the complete picture, it is necessary to understand the twin concepts of acceptance and confidence thresholds. Again, because the vocabulary of accuracy originated within the context of ICR, the ICR model of accuracy can yield insights into how IWR accuracy is expressed. In fact, IWR accuracy often is translated into its ICR equivalent in order to establish a rational pricing model.

During recognition, an ICR engine assigns a confidence value to each and every instance of character classification - usually on a scale of 0 to 100 - that indicates the degree of certainty that a neural-network-based ICR engine associates with each of its character choices. Although a confidence value does not, strictly speaking, represent the individual probability that a particular instance of character classification is correct, for all practical purposes it can be used like one by setting up confidence thresholds, either on a per-character or on a per-field basis.

Thresholds are barriers that ICR results must overcome in order not to be examined for accuracy by a human operator. The percentage of characters or fields that overcome a confidence threshold, when compared to the total number that tried to, is known as the acceptance rate.



Thus, the accuracy rate is the percentage of characters that are identified correctly by the ICR engine compared to the total number of accepted characters that cross the pre-set confidence threshold, including those characters that are wrongly identified. An incorrectly classified character that manages to somehow exceed a field confidence threshold is called a substitution error; if it is not discovered at the data validation level, say, by comparing it against a dictionary or other list of terms, it will pass through the system undetected. The number of substitution errors relative to the number of accepted characters is called the substitution error rate, and it is the most important index used to characterize recognition accuracy, regardless of whether it is reported on a character or field basis.

Using a similar process, a set of analogous relations can be determined with respect to IWR, albeit on a field level. Thus, with respect to IWR, the number of fields containing substitution errors relative to the number of accepted fields would be called the field substitution error rate. It is also true that, through analysis, the field substitution error rate can be analytically reduced to and expressed as, a per-character rate.

For an example, if 25 characters out of 1000 that crossed a confidence level were in reality substitution errors, then the overall character substitution error rate would be 25/1000 or .025, which is 2.5%. It gets a bit tricky, however, when the same data is expressed at the field level. For example, suppose that the 1000 characters are distributed over 200 data fields composed of 5 characters each. The same 25 errors could be distributed over only 5 of the fields (5 per-character substitution errors per field) or over 25 fields (1 character substitution error per field). The gross character substitution error rate would remain the same in both instances. The field substitution error rate, however, could vary dramatically. On the low end, it might be five fields out of one thousand, which is 5/200 for a 2.5% field error rate. On the high end, it could be twenty-five fields out of one thousand, which is 25/200 for a 12.5% field error rate. It is possible for the actual rate to be any figure in between, depending upon the empirical distribution of substitution errors across fields. It would vary according to the IWR application and writer population.

Field Level Accuracy and the Useful Work Criterion

To speak of field substitution rates is to be reminded that high character-level accuracy does not by itself ensure recognition success. For example, accurately recognizing nine out of ten digits in a telephone number means that 90% of the characters are correctly classified - an "A" in any teacher's book - but it also produces a wrong number and therefore does not do any useful work for the user. Accordingly, standards of field-level accuracy are required to ensure that IWR does useful work, a principal that, when applied, is appropriately called the useful work criterion. Of course, the useful work criterion can be enforced by setting high field-level confidence thresholds. Through a process of trial-and-error, end users of IWR can discover the confidence threshold that allows the fewest field substitution errors. Since the threshold is operator-controlled, the way to optimize IWR accuracy is to adjust the threshold setting so that it yields a high acceptance rate while simultaneously producing a substitution error rate comparable to those of human operators.

Confidence thresholds provide an IWR vendor with a means of pricing their software rationally and fairly. For example, with A2iA's FieldReader software, the customer pays only for recognition success based upon a specified threshold value. In other words, once the confidence thresholds are set, the customer pays a FieldReader license fee that counts only the words (expressed on a per-character basis) that score above the confidence threshold. The FieldReader license fee does not apply to rejected (below threshold) results, because paying for rejects would constitute a form of double payment. That is, the user would be paying the price for unacceptable recognition results in addition to the cost of manually correcting those same fields. How this pricing scheme might play out is depicted in a hypothetical IWR cost illustration displayed on page five.

The Cost of Correcting an Error

The further downstream in the data entry process that an error goes unnoticed, the costlier it is to repair. In the financial services industry in particular, it is critical to eliminate errors early on. When errors go undetected, the likelihood is that the customer will discover the error - under circumstances that are usually far from pleasant. At that point, the financial institution might find that the consequential damages of the error are far greater than just suffering the embarrassment of delivering poor customer service.

For example, take a stock market transaction, where an undetected error could translate into a wrong order that isn't discovered until the broker who placed the order receives confirmation of his transaction. Depending on how the stock market behaved in the meantime, if the market had turned the wrong way, the erroneous transaction could end up costing the broker-dealer thousands of dollars to reverse. Errors of this kind underscore the importance of the cost of an error. Of course, a user should make every effort to hunt down errors and correct them before they become part of a financial transaction that wreaks havoc on a customer.

The best strategy, of course, is to nip errors in the bud, i.e., prevent them from happening at all. One way to discover errors is to conduct an offline search for them by reading the batch error report and using it to locate the erroneous document in the files.

This cost is illustrated by the sidebar, which assumes (1) a data entry operator rate of 10,000 keystrokes per hour, (2) operator wages of $15 per hour or $.25 per minute, and (3) that it takes 20 minutes to find the wrong file plus a minute for a key entry operator to correct the error manually. Based upon these assumptions, the cost of finding an error offline and correcting it turns out to be $5.25.

When this figure is compared to the cost of manually correcting an are dramatic: the ratio of the former to the latter is 3500 to 1! When the offline error correction cost is compared to the cost of an IWR-corrected error, the cost ratio is even more disproportionate: it comes out to $5.25/.0006 or 8750 to 1!

The Cost of Correcting an Error

The following example illustrates the cost of correcting a single character keying error manually as compared to the IWR cost.

Assumptions:

Average keying rate = 10,000 keystrokes per hour (KPH).

Data entry operator costs = $15 per hour or $.25 per minute.

Average time to read batch error report and locate erroneous document from the files = 20 minutes.

Time to reenter correction transaction detected off-line = 1 minute.

Cost of IWR correction = .0006 per character.

Calculations:
Cost to manually correct single character error detected online: **Cost = 1 keystroke/ 10,000 KPH X $15 per hour = $.0015**

The cost to correct an error detected offline: **Cost = (20 min + 1 min) x $.25 per min = $5.25**

The ratio of the manual cost of offline error detection & correction versus manual online error detection & correction: **Ratio = $5.25 / $0.0015 = 3500 to 1**

The cost ratio of manual offline correction compared to IWR correction: **Ratio = $5.25 / .0006 = 8750 to 1**

Using IWR in a Double-Pass Environment

Clearly, if the risk of consequential harm caused by an error is so great as to require the highest attainable accuracy rate, then the least expensive way to ensure it is to use an online method of preventing errors from occurring. A popular and effective means of doing this is to key every field twice using two online operators working either serially or simultaneously together, and then compare the results for a match. Whenever the two answers match, the data is automatically sent to its destination repository. If they do not, the recognition results, along with images of the questionable fields, are sent to an operator to arbitrate and correct. So as not to break the key operator's rhythm, the arbiter is usually a third party. To save labor costs, an IWR engine can be used instead of a second data entry operator with no appreciable decline in overall accuracy.

Double pass or double key verification, if done properly, is capable of producing data entry accuracy of 100%. At a single operator cost of $15.00 per hour, here is how the cost of the operation is figured. First of all, a field is defined as 100% accurate when the operator results exactly match. Since a human operator yields 99.5% per-character accuracy with a complimentary substitution error rate of .5%, then a second data entry operator who is keying the same set of characters independently will key 99.5% of the errors from the first operator with 99.5% per-character accuracy. Those corrected errors will show up as mismatches, and so will the substitution errors that go uncorrected, because they will not match up with the other operator's results, either. So, each operator, regardless of which operator gets it right, will be responsible for mismatches equal to (100% - 99.5%) or .5% of the total number of characters keyed, which means that the number of mismatches from both operators will equal 1% of the total. At 10,000 characters per hour, that number would be 100. At a cost of $.0015 per character for the arbiter, that equates to $.15 an hour. But more significantly, because this method catches all operator mismatches, it means that all possible errors have been captured for final validation. Add that to the cost of two full-time operators, each at $15.00 per hour, and the total is $30.15 per hour to obtain virtual 100% accuracy.

Why is the 100% accuracy figure virtual and not actual? Theoretically, the arbiter occasionally makes an error - at the per character rate of .5% or one error for every 200 mismatched characters that he/she keys. That means that the net accuracy for two operators plus an arbiter is 19999/20000, which is 0.99995 or 99.995%.

Another way of looking at it is to note that before the double-pass system, the old system generated 100 substitution errors for every 20,000 characters keyed. Implementing the double pass system drops the number down from 100 substitution errors in 20,000 to only 1 in 20,000. Eliminating 99 errors out of 100 means that the number of substitution errors passing through the system has declined by 99%! Replacing one of the key operators with an IWR engine provides similar accuracy at a nice discount.

A hypothetical illustration of how IWR works in a double verification system is illustrated on the next page. Although the A2iA FieldReader software recognizes characters on a field basis, for purposes of illustration, the statistics are calculated at the character level. Add to the above cost parameters an assumed per-character substitution error rate of 5% and a hypothetical cost of $.06 for every character that the IWR system accepts above the threshold. The cost differential between an all-manual double pass system and a double-pass system that replaces a human operator with IWR software can then be computed by making the appropriate cost substitution and then running the numbers. The results are shown on the next page.

Minimizing the Costs of Maximizing Data Entry Accuracy

Computing the cost of double key entry using an arbiter plus an IWR engine substituted for a human runs as follow. First, the cost of manually entering all the data once is $15.00 an hour. Instead of entering that number again for the human, we enter the cost of the IWR engine. The cost in the example is .0006 per character. The substitution error rate of IWR engine is 5%. The acceptance rate in the example is .68% of all characters in all fields. Taking an example of 10,000 characters, that means 6800 characters would be compared by the IWR engine with the human operator performance, for a cost of 6800 X .0006 = $4.08. The rest would be rejected by the IWR engine and so do not incur a cost. The number of mismatches between the human and the IWR system would be equal to the 5% IWR substitution errors (6800 X .05 = 340 characters) that didn't match up with the human operator plus the human substitution errors caught by the IWR system (6800x.005 = 34 characters). That yields a total of 374 characters that will be keyed by the arbiter. The cost of the arbiter keying these characters would be 374 X .0015 or $0.56 per 10,000 characters. The total cost per hour would be $15.00 + $4.08 + $.56, or $19.64 per hour.

Calculating the accuracy rate per 10,000 characters runs as follows. The first step is determining the net number of errors per 10,000 characters, and then comparing it with the cost and accuracy of what it replaces. The net number of errors is computed as follows. For every 10,000 characters, all errors have been eliminated from 6800 characters, with the exception of the arbiter errors, which would be .005 X 374 = 1.87. On the remaining 3200 characters that are manually keyed by the data entry operator, the number of errors will be .005 X 3200 = 16 errors. Therefore, the total number of errors per 10,000 characters is 17.87, which means that 9982.13 characters were keyed correctly. Therefore the accuracy rate is 9982.13/10000 = .998213, which is 99.8213% accuracy. The comparison among one human operator, versus two, versus one operator plus one IWR engine, arranged in terms of ascending cost is as follows:

	Single Pass MDE only	MDE plus IWR plus Arbiter	Two-pass MDE plus Arbiter
Number of Characters:	10,000	10,000	10,000
Cost per 10,000 characters:	$15.00	$19.65	$30.15
Number of substitution errors:	50	17.87	0.5
Number chars. Correct:	9950	9982.13	9999.5
Per-character accuracy rate:	99.500%	99.821%	99.995%

This illustration shows that, by adding an IWR engine to the system, the number of errors per ten thousand is diminished by 32 characters, from 50 to 18 characters for a cost of $4.64 per 10,000 characters, or 15 cents a character. Virtually eliminating the last 18 errors down to .5 errors per 10,000 characters costs an additional $10.51, or 60 cents a character. It's a question of marginal utility and the potential financial consequences of an error, depending upon a specific application. Clearly, since the IWR engine, in conjunction with the human operator, virtually eliminates all of the errors within the range of characters it accepts, the IWR acceptance rate is the key to optimizing accuracy using the MDE plus IWR approach. The higher the acceptance rate, the better the results, depending on how high the substitution error rate climbs as the acceptance rate goes up.

The spreadsheet on the following page illustrates how this same scenario plays out when the scale is increased to real life proportions in an operation where one million forms are being processed. For the sake of simplicity, only two fields are illustrated, an "item name" field and an "item number" field. The savings that occur are impressive: $27,261, which represents a drop of over 33%. Again, the decision on the size of the percentage of accuracy improvement that is practical will vary from one application to another.
Those data entry operations that require an accuracy rate that is greater than the human standard of 99.5% and that desire the biggest improvement in accuracy per dollar, should begin by adding an IWR engine to their system.

Double-Pass Verification: IWR Replaces One Operator

Name of field	Total number of expected characters in field	Cost MDE. (1 Pass)	Cost of characters above Threshold	IWR + MDE Total Cost of Arbiter Entry	Total Cost MDE W/2 Passes	Total Cost of IWR & MDE (1 pass)	**Total Cost Savings using IWR**
Item Name	10	$15,000	$4,080	585	$30,150	$19,665	**$10,485**
Item Number	16	$24,000	$6,528	$936	$48,240	$31,464	**$16,776**
Total	26	$39,000	$10,608	$1,521	$78,390	$51,129	**$27,261**

Assumptions:

MDE Price p er character = .0005

IWR Price per character = .0006

Number of documents = 1,000,000

IWR Acceptance rate = .68

IWR Sub. Error Rate = .05

Applications Guidelines for IWR Implementation

Setting up an IWR application can be done as easily by an end user as it can by a developer or an integrator. A2iA's graphical user interface is highly intuitive; its user-friendliness enables installations to run smoothly, including those that require the creation of an application-specific dictionary. At the same time, A2iA's developer API permits software programmers and professionals to integrate its software recognition modules into the most complex applications.

IWR is germane to any number of forms processing applications that involve recognition of imaged data. However, IWR is best suited for high-volume applications where ICR has a high failure rate; or in those applications where ICR generates a large number of exceptions because of the regular appearance of fields containing handwriting or run-on hand print. IWR opens up possibilities for applications that in the past would have been impractical to consider because of their apparent illegibility, like traffic tickets and police accident reports. The major form types suggested by the parameters just described are listed below. They define applications for which A2iA Fieldreader is an excellent solution.

Semi-structured forms, such as invoices and purchase orders, cause trouble for ICR engines because they require the capability of finding key fields that could be either printed or handwritten. The field location and data capture problems related to these documents are further compounded by numerous appearances of different amount and date field designations. As previously described, IWR can accurately recognize them with the aid of an embedded list of aliases.

Surveys and questionnaires often rely on the use of open fields, which appear in forms where the instructions ask members of the survey group to answer certain questions in writing with a word or phrase. IWR can accurately recognize these handwritten words with the aid of a dictionary of verbal expressions most frequently used by respondents.

Old public records are full of property titles, land deeds, conveyances, court cases, personnel records, library cards and other archival documents that are typeset in outdated seraphic fonts and that are composed of numerous fields that contain handwritten or calligraphic data. IWR can recognize and index a substantial percentage of these documents during backfile conversion. Government revenue-producing forms such as hunting and fishing licenses, vehicle registration forms, and traffic tickets, all require manual data entry because they can be filled out in handwriting. With all these forms, and especially the latter two groups, IWR aliases come in particularly handy; for example, "Chevy", "Chevrolet", and "Chev." are aliases that refer to the same make of car.

Order entry forms from mail order organizations frequently contain open fields in which a customer can write in the number of items they are ordering when it exceeds the number of choices that can be checked on the form. Other order forms allow customers to write catalog codes and descriptions in blank fields. IWR can accurately identify and validate the handwritten descriptions and amounts and then process them.

Consumer letters of complaint, which commonly are handwritten or hand printed, play an important role in customer service, especially with credit card providers and mail order companies. At the very least, IWR could use aliases to group complaints by keyword frequency and index them; the likelihood is that IWR could recognize a significant percentage of this correspondence.

Major Benefits of IWR

The major benefits of using IWR include classifying semi-structured documents, recognizing difficult handwriting and run-on hand print on forms, and otherwise recognizing data fields on forms that are unsuitable for ICR. These capabilities lower the cost of data entry on documents that present the most difficult recognition problems; moreover, any business cycle that is dependent on data entry is shortened in the process.

If the business process involved is mission-critical, the results can yield a "ripple effect" that permits information to be interpreted and distributed faster across the enterprise. Faster data distribution also means quicker response time regarding customer requests, which means improved customer service.

By removing the routine data-entry tasks that IWR is able to automate, the use of confidence levels allows data-entry workers to concentrate on more difficult exception items. More time is now available for workers to allocate to data perfection and to verify that work is completed on time. Moreover, by reducing the amount of keying, workers are less likely to develop carpel tunnel syndrome, repetitive-stress injury, or other ergonomically-related inflictions associated with data entry activities.

IWR can make data-entry service bureaus more productive by allowing them to create new markets and enable new lines of business. In fact, the superior recognition capabilities of IWR enable A2iA Fieldreader to automate data entry to domestic service bureaus at a per-character cost so low that they can successfully compete against offshore service bureaus located in a country like India or China. In many instances, IWR could be used as a substitute for off-shoring data entry tasks. In others, it is more economical to supplement IWR with offshore data entry correction of rejected fields. The method used will depend upon the economics of the particular recognition application.

While IWR does not occupy a prominent place on the radar screen of the CEO, it nonetheless can yield benefits that can outstrip applications and technologies that are. IWR, when applied successfully, is a powerful cost-reduction tool that at the same time can upgrade the use of human and corporate resources.

IWR is a low risk, high-tech solution that, when applied successfully, greatly improves data entry throughput, truncates information processing lifecycles and improves business processes throughout the enterprise. It yields tremendous cost savings that ultimately create enormous productivity gains for the companies that adopt it. In the end, the business benefits of IWR drop straight to the bottom line. They end up creating a competitive advantage for the business user - one that can be crucial for survival in today's data-intensive business environments.

Arthur Gingrande, Jr. is a nationally acclaimed expert on character recognition, electronic forms, automated forms processing, and mail classification. He is a certified Image Capture Professional and an original partner of IMERGE Consulting (www.imergeconsult.com), a firm specializing in document-centric, work process applications and technologies. He is also one of the founders of Symbus Technology (now called Captiva Software), an ICR development firm and systems integrator, as well as the former Director of Marketing and Business Development for Nestor, Inc., a leading provider of neural networkbased, pattern recognition software.

Mr. Gingrande was the first imaging editor of ISIT.com (now Vertical Markets IT Group), and is currently the editor and publisher of Contemplor, a newsletter dedicated to document management and forms recognition technologies. He has participated in scores of conferences and trade shows - put on by organizations such as AIIM, TAWPI, ARMA, ScanTech, USPDI and BAI - as a guest speaker, a panelist, a pundit, and an industry commentator. At present, he is President of the New England Chapter of The Association for Work Process Improvement (TAWPI). He is also a member of the Editorial Board of AIIM.

Since 1991, over 250 of Mr. Gingrande's articles, essays and papers have appeared in various trade periodicals such as AIIM e-DOC Magazine, Inform, KM World, Business Solutions, Integrated Solutions, Imaging Business, and VAR Business. Topics include workflow, document management, electronic imaging, COLD/ERM, ICR/OCR, eforms, e-business, CRM, knowledge management, wireless communications, processing unstructured documents, and enterprise content management. He is the author of numerous white handbooks and user guides, including Using Forms Automation to Boost Enterprise Productivity, Forms Automation - from ICR to Electronic Forms to the Internet, Processing Unstructured Documents - Challenges and Solutions, and Technology Convergence, Document Management, and E-commerce, all published by AIIM. He also wrote Cost-Justifying an ICR Solution, and Measuring and Improving Data Entry Productivity, both published by TAWPI.

Arthur Gingrande Jr. graduated with a B.A. in English from Wesleyan University in Middletown, CT, and received his Master of Arts in Philosophy from the American University in Washington, D.C. His practice is located in Lexington, Massachusetts. He may be contacted by telephone at 781-258-8181, or by e-mail at arthur@imergeconsult.com.